Postprint of Intrusion Detection Algorithm Based on Deep Sequential Weighted Kernel Extreme Learning Machine

Authors: Wang Yang, Wu Zhongdong, Zhu Jing

Date: 2019-01-03T00:00:00+00:00

Abstract

To address the challenges of network intrusion detection posed by massive, multi-source, heterogeneous data with imbalanced distributions, and the limitation of traditional deep learning algorithms in their inability to update output weights online in response to real-time intrusion scenarios, an intrusion detection algorithm based on Deep Sequential Weighted Kernel Extreme Learning (DBN-WOS-KELM) is proposed. The algorithm first employs a Deep Belief Network (DBN) to learn from historical data, accomplishing feature extraction and dimensionality reduction of the raw data, and subsequently utilizes a Weighted Sequential Kernel Extreme Learning Machine for supervised learning to perform intrusion identification, thereby combining the capacity of DBN for extracting abstract features with the rapid learning capability of Kernel Extreme Learning Machines. Finally, simulation experiments conducted on a subset of the KDD99 dataset demonstrate that the DBN-WOS-KELM algorithm enhances the recognition rate for small-sample attacks and is capable of online updating of output weights according to actual conditions, achieving superior training efficiency.

Full Text

Preamble

Vol. 37 No. 3
Application Research of Computers
ChinaXiv Cooperative Journal

Intrusion Detection Algorithm Based on Deep Sequence Weighted Kernel Extreme Learning

Wang Yang, Wu Zhongdong, Zhu Jing
(School of Electronic & Information Engineering, Lanzhou Jiaotong University,

Lanzhou 730070, China)

Abstract: To address the challenges of massive multi-source heterogeneous network intrusion detection with imbalanced data distribution, and the inability of traditional deep learning algorithms to update their output weights online according to real-time intrusion situations, this paper proposes an intrusion detection algorithm based on deep sequence weighted kernel extreme learning (DBN-WOS-KELM). The algorithm first employs a Deep Belief Network (DBN) to learn from historical data, completing feature extraction and dimensionality reduction of raw data. It then utilizes a weighted sequence kernel extreme learning machine for supervised learning to accomplish intrusion identification, combining DBN' s capability to extract abstract features with the fast learning ability of kernel extreme learning machines. Finally, simulation experiments were conducted on a subset of the KDD99 dataset. Experimental results demonstrate that the DBN-WOS-KELM algorithm improves the recognition rate of small-sample attacks and can update output weights online according to actual conditions, achieving higher training efficiency.

Keywords: deep belief network; sequence learning; kernel extreme learning; sample weighting; intrusion detection

CLC number: TP393.08
doi: 10.19734/j.issn.1001-3695.2018.08.0653

0 Introduction

Intrusion detection technology constitutes an indispensable component of information network security. With the continuous advancement of artificial intelligence technology, deep learning algorithms offer advantages of higher recognition rates and lower false positive rates compared to traditional machine learning-based intrusion detection algorithms, finding widespread application in the intrusion detection domain. Gao et al. [1] proposed an intrusion detection algorithm based on deep belief networks, Ambusaidi et al. [2] introduced a hybrid intrusion detection algorithm based on DBN-SVM, and Lu and Yang [3,4] presented hybrid intrusion detection models based on deep learning, all achieving experimental results superior to traditional machine learning algorithms in terms of accuracy and false positive rates. However, current learning algorithms have not considered the imbalance in the distribution of historical network intrusion data used for training, focusing solely on high detection rates and low false positive rates. This oversight results in small-sample attack categories being mostly misclassified as large-sample attack categories, yielding low detection accuracy for small-sample attack types.

To address such imbalanced data classification problems, Zong et al. [5] proposed a weighted extreme learning machine algorithm for handling imbalanced data. Mirza, after studying the advantages of sequence learning, introduced

weighted online sequential extreme learning machine algorithms for imbalanced data learning and weighted sequence kernel extreme learning algorithms [6,7], with experimental results showing significant improvement in small-sample data recognition. This paper focuses on the multi-source heterogeneous nature of network intrusion data, the imbalanced distribution characteristics of various attack categories, and the inability of traditional deep learning algorithms to update their output weights online based on real-time intrusion data. Through in-depth research on deep belief networks and sequence kernel extreme learning machines, we propose an intrusion detection algorithm based on DBN-WOS-KELM, which fully leverages DBN' s capability to extract data features and the generalization ability of sequence kernel extreme learning machines. Finally, effective evaluation through a subset of the KDD99 dataset demonstrates that our algorithm not only improves the detection rate for small-sample attacks but also enables online updating of classifier parameters according to actual conditions, further enhancing training efficiency.

1.1 Overall Architecture

This paper proposes the DBN-WOS-KELM intrusion detection algorithm, with the overall framework illustrated in [Figure 1: see original paper]. The specific steps are as follows:
a) Data preprocessing: Convert character features in network data into corresponding binary data, then normalize to [0,1].
b) DBN abstract feature extraction: Perform unsupervised pre-training of RBMs followed by global BP fine-tuning to reduce the dimensionality of network data.
c) WOS-KELM classifier for intrusion identification: Apply labels to the dimensionality-reduced data as reliable data, perform sample weighting, and employ KELM sequence learning. After training, replace BP as the classifier.
d) DBN-WOS-KELM intrusion identification: Apply sample weighting to dimensionality-reduced network historical intrusion data for intrusion detection and type identification.

To achieve global optimization of the entire DBN network, a BP network utilizes a small amount of reliably labeled data to fine-tune the parameters of each RBM layer. The predicted values are compared with actual labels to obtain errors, which are then propagated to every layer. The specific steps are shown in Algorithm 2.

Algorithm 2: Backpropagation Algorithm
Input: Pre-trained DBN parameters, maximum iteration count.
Training samples <x, t>, learning rate .
Output: Fine-tuned DBN parameters.
Training phase: For each sample, compute DBN' s reconstructed output, backpropagate errors.
For each output unit, compute error: = (t - y) y (1 - y).
For each hidden layer unit, compute error: = (W^T _next) h (1 - h).

Update parameters: W = W + h^T, b = b + .

1.2 DBN Feature Dimensionality Reduction

A Deep Belief Network is a generative deep structure composed of multiple layers of RBMs and one BP network, first proposed by Hinton [8], as shown in step b) of [Figure 1: see original paper]. Its training process can be decomposed into two steps:

1) RBM Pre-training
   The RBM structure is shown in [Figure 2: see original paper], consisting of an input layer and a hidden layer. The energy function is defined as:

$$E(v,h) = -b^T v - c^T h - h^T W v$$

   where b is the bias from input layer to hidden layer, c is the bias from hidden layer to input layer, W is the weight matrix connecting visible and hidden layer nodes, and v and h represent the states of neurons in the two layers respectively. The RBM training algorithm is the contrastive divergence learning algorithm proposed by Hinton [9][10], with the process as follows:

Algorithm 1: Contrastive Divergence Learning Algorithm
Input: Training samples; number of hidden layer neurons; learning rate ; maximum iterations.
Output: RBM parameters.
Training phase: Randomly initialize parameters = {W, b, c}.
For all hidden units: P(h=1|v) = (b + Wv), sample h {0,1}.
For all visible units: P(v' =1|h) = (c + W^T h), sample v' {0,1}.
For all hidden units: P(h' =1|v') = (b + Wv'), sample h' {0,1}.
Update parameters: ΔW = (P(h=1|v)v^T - P(h' =1|v')v' ^T), Δb = (v - v'), Δc = (P(h=1|v) - P(h' =1|v')).

1.3 WOS-KELM Classifier

Weighted Online Sequential Kernel Extreme Learning Machine (WOS-KELM) [7] combines weighted sequential learning algorithms with Weighted Kernel Extreme Learning Machine (WKELM) [5], proposed by Shuya Ding et al. Similar to Online Sequential Kernel Extreme Learning Machine (OS-KELM) [11], the training process is divided into an initialization phase and a sequential learning phase.

1) Initialization Phase
   The initialization phase of WOS-KELM is similar to the WELM learning algorithm. First, select initial training data and determine the hidden layer initialization output matrix. The initial output weight is calculated

as:

$$\beta^{(0)} = (H_0^T W_0 H_0 + CI)^{-1} H_0^T W_0 T_0$$

where K is the kernel function, T_0 is the sample label matrix, C is the regularization coefficient, and W_0 is the weight matrix for the first batch of training samples. The calculation method for W_0 is given in literature [7]:

$$w_i = \frac{1}{N} \cdot \frac{N}{N_j}$$

where N represents the total number of training samples, N_j represents the number of training samples belonging to class j, and N is the number of samples in class j.

2) Sequential Learning Phase
When new training data participates in training, the updated kernel extreme learning machine output matrix is:

$$\beta^{(k+1)} = \begin{bmatrix} K_{k+1}^T W_{k+1} K_{k+1} + CI & K_{k+1}^T W_{k+1} \\ W_{k+1} K_{k+1} & W_{k+1} \end{bmatrix}^{-1} \begin{bmatrix} K_{k+1}^T W_{k+1} T_{k+1} \\ W_{k+1} T_{k+1} \end{bmatrix}$$

where W_{k+1} = diag(w_{k+1,1}, ···, w_{k+1,N_{k+1}}), w_{k+1,i} = m/k_i, N_{k+1} is the number of samples in batch k+1, and k_i is the class index of sample i.

2.1 Dataset Preprocessing

The experiment adopts the KDD99 dataset [12], which categorizes anomalies into four major types: DOS, R2L, U2R, and Probe, containing 39 attack methods. The training set includes 22 attack methods, while the test set contains 17 additional attack methods not present in the training set (these were not used in our experiments). The dataset has 41-dimensional features, including both character and numeric types. Preprocessing is performed before training and detection: first, character data is converted into binary vectors. For example, label types normal, DOS, R2L, U2R, and Probe are represented as [0,0,0,0,1], [0,0,0,1,0], [0,0,1,0,0], [0,1,0,0,0], and [1,0,0,0,0] respectively. After character mapping, data normalization is applied to scale each dimension to between 0 and 1 using the formula:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

2.2 Experimental Evaluation Criteria

Traditional intrusion detection algorithms typically emphasize overall high accuracy and low false alarm rates while neglecting detection of small-sample

attacks. In addition to using Accuracy (AC) and False Alarm rate (FA) as evaluation standards for various attack detections, this paper also employs Geometric Mean (G-mean) [13] as an overall evaluation standard, as shown in equations (18)-(20):

$$AC = \frac{TP + TN}{TP + TN + FP + FN}$$

$$FA = \frac{FP}{FP + TN}$$

$$G\text{-mean} = \sqrt[|S|]{\prod_{j=1}^{|S|} \frac{TP_j}{TP_j + FN_j}}$$

where TP represents correctly classified normal samples, TN represents correctly classified attack samples, FP represents normal samples misclassified as attacks, FN represents attack samples misclassified as normal, N_j represents the total number of class j attack samples, TP_j represents correctly detected class j attacks, and |S| represents the number of sample categories (5 in our experiments).

2.3 Experimental Parameter Settings

In the proposed model, DBN serves as the dimensionality reduction component, and determining its network depth is crucial. The experiment preset five DBN structures (where the number indicates network layers). The classifier uses a BP network with node parameters shown in . By extracting 20% from the KDD99 training set and 10,000 records from the test set, detection results for various attacks were obtained. Based on equation (20), the G-mean values under different network depths were calculated. Following the principle of selecting the depth with highest G-mean, DBN4 was chosen (as shown by the dashed line in [Figure 3: see original paper]), with specific detection results presented in [Figure 3: see original paper].

KELM parameters include the kernel function, which is a Gaussian kernel function defined as:

$$K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2)$$

where $\gamma > 0$.

TABLE:1 DBN Parameters

2.4.1 DBN-WKELM Experiment

To verify detection performance under different training set distributions, this experiment extracted four training subsets with different sample distributions from the training set and 5,000 records from the test set. The detection performance of three algorithms—DBN, DBN-KELM, and the improved DBN-WKELM—were compared for five attack types, along with their G-mean values. The data distribution is shown in .

TABLE:2 Training Data Distribution

Detection results for each attack sample are presented in through , with G-mean values calculated according to equation (20) and shown in [Figure 4: see original paper]. Experimental results demonstrate that DBN-KELM achieves higher detection rates than the original DBN algorithm. The weighted DBN-WKELM algorithm, while minimally sacrificing detection rates for large-sample attack types, significantly improves detection rates for small-sample attacks. Across the four training subsets, the G-mean values of DBN-WKELM exceed those of the other two algorithms, compensating for the traditional intrusion detection limitation of high recognition rates for large-sample attacks but low rates for small-sample attacks.

TABLE:3 Test Results of Dataset 1
TABLE:4 Test Results of Dataset 2
TABLE:5 Test Results of Dataset 3
TABLE:6 Test Results of Dataset 4

2.4.2 DBN-WOS-KELM Experiment

This experiment extracted 8,000 records from the KDD99 training set, divided into 8 data chunks, and 5,000 records from the test set (excluding unknown attack types). During training, each data chunk was processed through DBN dimensionality reduction and then fed into the sequential kernel extreme learning machine (WOS-KELM) for training, sequentially updating the kernel extreme learning machine' s output weights to simulate real-world online intrusion detection system updates. Each data chunk contained equal amounts of data, with specific distribution shown in . Due to the small U2R quantity in the training set, a self-replication method was applied.

TABLE:7 Distribution of Batch Data

[Figure 5: see original paper] illustrates the detection rate changes for each attack type and the variation of G-mean as training batches increase. Results show that as training progresses, detection rates for various attack types gradually stabilize, with G-mean values increasing sequentially but at a decreasing rate, indicating classifier stabilization. This validates the effectiveness of the proposed DBN-WOS-KELM algorithm.

[Figure 6: see original paper] compares training time curves between DBN-WOS-

KELM and DBN-WKELM under identical data volumes. Experimental results demonstrate that DBN-WOS-KELM exhibits significantly superior training efficiency compared to DBN-WKELM.

3 Conclusion

This paper addresses the challenges of massive multi-source heterogeneous network intrusion detection with imbalanced data distribution and the inability of traditional deep learning algorithms to update output weights online based on real-time intrusion data. We propose a deep sequence weighted kernel extreme learning intrusion detection algorithm (DBN-WOS-KELM) that combines the advantages of DBN feature extraction and KELM fast learning. Through sample weighting, the algorithm resolves training issues under data distribution imbalance, improving recognition rates for small-sample attacks while minimally reducing recognition rates for large-sample attack categories. Moreover, it enables online updating of output weights when new training data arrives. Experiments on a subset of the KDD99 dataset demonstrate that DBN-WOS-KELM achieves high recognition rates for various attack types, can update its output matrix online according to actual conditions, and exhibits superior training efficiency compared to DBN-WKELM under equivalent data volumes. However, this study only validated the algorithm's feasibility using a subset of the KDD99 dataset; future work will consider applying the algorithm to real-world intrusion detection scenarios.

References

[1] Gao Ni, Gao Ling, He Yiyue, et al. Intrusion detection model based on deep belief network [J]. Journal of Southeast University: English Edition, 2015, 31(3): 339-346.
[2] Ambusaidi M A, He Xiangjian, Nanda P, et al. Building an intrusion detection system using a filter-based feature selection algorithm [J]. IEEE Trans on Computers, 2016, 65(10): 2986-2998.
[3] Yang Kunpeng. Intrusion detection based on deep learning [D]. Beijing: Beijing Jiaotong University, 2015.
[4] Lu Yujing. Research on intrusion detection algorithm based on deep belief network [D]. Shijiazhuang: Hebei Normal University, 2016.
[5] Zong Weiwei, Huang Guangbin, Chen Yiqiang. Weighted extreme learning machine for imbalance learning [J]. Neurocomputing, 2013, 101: 229-242.
[6] Mirza B, Lin Zhiping, Toh K A. Weighted online sequential extreme learning machine for class imbalance learning [J]. Neural Processing Letters, 2013, 38(3): 465-486.
[7] Ding Shuya, Mirza B, Lin Zhiping, et al. Kernel based online learning for imbalance multiclass classification [J]. Neurocomputing, 2018, 277: 1-13.
[8] Hinton G E, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets [J]. Neural Computation, 2006, 18(7): 1527-1554.
[9] Hinton G E. Training products of experts by minimizing contrastive diver-

gence [J]. Neural Computation, 2002, 14(8): 1771-1800.
[10] Hinton G E. A practical guide to training restricted Boltzmann machines [M]//Neural Networks: Tricks of the Trade. Berlin: Springer, 2012: 599-619.
[11] Deng WanYu, Ong Y S, Tan P S, et al. Online sequential reduced kernel extreme learning machine [J]. Neurocomputing, 2016, 174: 72-84.
[12] Dhanabal L, Shantharajah S P. A study on NSL-KDD dataset for intrusion detection system based on classification algorithms [J]. International Journal of Advanced Research in Computer and Communication Engineering, 2015, 4(6): 446-452.
[13] Al Helal M, Haydar M S, Al Mostafa S M. Algorithms efficiency measurement on imbalanced data using geometric mean and cross validation [C]//Proc of International Workshop on Computational Intelligence. Piscataway, NJ: IEEE Press, 2016: 110-114.

Note: Figure translations are in progress. See original paper for figures.