A Deep Auto-Encoder based LightGBM Approach for Network Intrusion Detection System

Kun Mo[1], Jian Li[1]
[1]*School of Computer Science, BeiJing University of Posts and Telecommunications, Beijing, China*

Keywords: Intrusion detection systems, Deep auto-encoder, LightGBM, Cyber security, Multi-classification algorithm.

Abstract: With the development of the network in recent years, cyber security has become one of the most challenging aspects of modern society. Machine learning is one of extensively used techniques in Intrusion Detection System, which has achieved comparable performance. To extract more important features, this paper proposes an efficient model based Auto-Encoder and LightGBM to classify network traffic. KDD99 dataset from Lee and Stolfo (2000), as the benchmark dataset, is used for computing the performance and analyse the metrics of the method. Based on Auto-Encoder, we extract more important features, and then mix them with existing features to improve the effectiveness of the LightGBM (Ke et al., 2017) model. The experimental results show that the proposed algorithm produces the best performance in terms of overall accuracy.

1 INTRODUCTION

Network intrusion detection system takes an important role in cyber security. But with the development of internet, various forms of network attacks are emerging one after another. In the past few years, network criminals have been constantly renovating the attack means, in addition to the use of zero-day vulnerabilities, malicious mining extortion software has been rampant; DDoS attack has made a breakthrough in TB level, and the attack channel is increasingly changing; 53% medium-sized companies have suffered security attacks; industrial network has become the focus of illegal hacker attack. So the whole industry network security environment brings new challenges.

The Intrusion Detection System (IDS) is a system which can detect network traffic for suspicious activity (Bai & Kobayashi, 2003). IDS has made great progress in past 20 years that various fine-designed machine learning algorithms have been proposed to model network traffic and applied on commercial applications, such as SVM, XGBoost, LightGBM and CNN. For example, Aastha Puri and Sharma (2017) used a novel algorithm named SVM-CART to combine its output from SVM and CART. Currently, existing IDS algorithm techniques can fall into two categories. The first category is called traditional machine learning model which is easy to train with better interpretability. Based on the characteristics of KDD99 dataset and the existing experimental results, traditional machine learning is extremely suitable for IDS to detect malicious behavior. The second category is deep learning method, which tends to excavate potential time or space structure characteristics. Because of lack of sufficient theoretical support, the second category is like a black box. In recent years, deep learning has developed rapidly so that it has many mature application scenarios, e.g., Computer Vision (CV) (Umbaugh, 1997), Natural Language Processing (NLP) (Manning & Schütze, 1999), autopilot (Ribler et al., 1998). They are proven to be capable of extracting high-order features and the correlation between features or adjacent data. Inspired by this, we can apply deep learning techniques to the feature extraction of network traffic data. This paper aims to combine the advantages of the two categories to achieve higher accuracy. The performance was then evaluated on KDD99 dataset and compared with other proposed model.

In this paper, we adopt deep Auto-Encoder and LightGBM algorithms to construct model. There are two phases in this model. In the first phase, we applied the clean dataset into deep Auto-Encoder network. We only save the intermediate results as the part of input data of the second phase. In the second phase, we concatenate the previous result and the

Mo, K. and Li, J.
A Deep Auto-Encoder based LightGBM Approach for Network Intrusion Detection System.
DOI: 10.5220/0008098401420147
In *Proceedings of the International Conference on Advances in Computer Technology, Information Science and Communications (CTISC 2019)*, pages 142-147
ISBN: 978-989-758-357-5

clean dataset into new dataset. Then the new dataset is inputted in the LightGBM model. In Section II, we analyse the related work. In Section III, we introduce the detail of our design. In Section IV, we introduce raw data, data pre-processing and evaluation metrics. In Section V, we give the results of the experiment and compare performance of various methods. Finally, in Section VI, we conclude this paper.

2 RELATED WORK

The concept of intrusion detection system from a technical report submitted to the US Air Force by Anderson (1980), which details what is intrusion detection. The core of intrusion detection is to use existing computer technology to analyse and detect network traffic, and then take corresponding measures according to certain rules. After more than 30 years of development, the intrusion detection technology has achieved many exciting results (Aljawarneh, Aldwairi, & Yassein, 2018). The existing mainstream intrusion detection methods are based on different machine learning algorithms and typical neural network algorithms, such as support vector machine (SVM) (Mahmood, 2018), Naive Bayes Multiclass Classifier, DNN, CNN (Nguyen et al., 2018).

One of the earliest work found in literature used SVM with various Kernel functions and regularization parameter C values for the design of the model (Kim & Park, 2003). In its paper, Dong Seong Kim and Jong Sou Park used 10% of the KDD 99 training dataset for training and all test dataset for testing. As expected, the training data was divided into train set and validation set. Instead of k-fold cross validation, they optimized the model by repeatedly sampling training set randomly. It is worth noting that the experimental results are improved a lot by proper feature selection. The experimental results proved that SVM achieved a high accuracy in IDS.

Inspired by SVM model, Sungmoon Cheong proposed new model named SVM-BTA to improve SVM (Cheong, Oh, & Lee, 2004). He produced a novel structure which includes SVM and decision tree. This work built a binary decision tree which each node was a SVM classifier. Besides of this, Sungmoon Cheong produced a modified SOM to convert multi-class tree into binary tree. As expected, this method got took advantage of both the efficient computation of the tree architecture and high accuracy.

Deep learning has become more and more popular since researchers were satisfied with data and computation. Javaid et al. (2016) proposed a 2-level deep learning structure. In the first level, there is a self-taught learning model, or more specifically, a Sparse Auto-Encoder for a more expressive feature representation. Sparse Auto-Encoder can excavate more relationships between features and labels. The second level is a softmax regression classifier. Moreover, Farahnakian and Heikkonen (2018) recently have proposed a deep Auto-Encoder based approach for IDS. There are five Auto-Encoder stacked together in their model. Then they used a supervised learning algorithm to avoid overfitting and local optima. Finally, a softmax classifier is added to get the results.

In this paper, we proposed an deep Auto-Encoder and LightGBM based approach for improving IDS performance. Our main contributions are as follows:

Firstly, an Auto-Encoder model is added to discover efficient feature representations.

Secondly, our model concatenated the intermediate result of the deep Auto-Encoder and the clean dataset into new dataset so that we can avoid the loss of feature transformation and feature reduction. We employed a LightGBM model to classify data.

Finally, the performance of our model is evaluated by KDD-CUP'99 dataset. A series of experiments is conducted to explore the performance of different parameters.

3 DEEP AUTO-ENCODER BASED LIGHTGBM MODEL

3.1 Auto-Encoder

An Auto-Encoder is a deep learning model which uses a backpropagation algorithm to make the output value equal to the input value. It first compresses the input into a latent spatial representation and then reconstructs the output by this characterization. An Auto-Encoder includes two parts.

Encoder: This part compresses the input into a latent representation, which can be represented by the encoding function $h = f(x)$.

Decoder: This part can reconstruct the input from the latent representation, which can be represented by the decoding function $y = g(h)$.

Auto-Encoder is an unsupervised learning algorithm whose structure is consistent with BP neural network, but its objective function is different:

$$y_i = x_i \tag{1}$$

For a given dataset $X = \{x_1, x_2, x_3, \dots, x_n\}$, where there is a constraint $x_i \in R^m$, Auto-Encoder first maps it to a hidden representation h by the function denoted as

$$h = f(x) = sigmoid(W_i \cdot x + b_\mathrm{i}), \tag{2}$$

where $W_i \in R^{l_1 \times m}, b_\mathrm{i} \in R^{l_1}$, l_1 is the number of hidden units. In this function, 0077e use a sigmoid function as the activation function to get the nonlinear correlation between features:

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

In the decoding stage, the model maps the encoding result to a reconstructed feature $y \in R^{l_1 \times o_1}$ as

$$y = g(x) = sigmoid(W_o \cdot h + b_o) \tag{4}$$

In next stage, the parameters are optimized such that the error of this model is minimized. The traditional squared error is used frequently:

$$L(x, y) = \frac{1}{n}\sum_{i=1}^{n} \left|\left|x_i - y_i\right|\right|^2 \tag{5}$$

n is the number of training dataset.

Finally, back propagation algorithm is used to optimize parameters.

3.2 LightGBM

SVM, GBDT, XGBoost, etc., are used frequently in IDS, but they will become very slow during training phase. Besides, their efficiency and scalability are worrying when the number of features or data is so large. The reason is that when splitting a node, they need to scan all the data to estimate the splitting gain of all possible segmentation point for each feature to get obtain the maximum information gain. So we chose LightGBM algorithm. LightGBM mainly includes two improved algorithms: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB).

GOSS is proposed to filter most data and only keep a small amount of data. In GBDT, a negative gradient is used for fitting the residual. A sample with a gradient close to zero indicates that the sample has been training well, and the subsequent decision tree will focus on training samples with larger absolute gradient values. The improvement of the GOSS algorithm is to sort the gradients, then select the samples with larger gradients and randomly extract a part of the samples with smaller gradients, and make up a constant for the small gradients, thus greatly reducing the training data.

EFB algorithm can avoid unnecessary computation for zero feature values by bundling exclusive features. The so-called mutual exclusion means that in the feature space, there is only one non-zero values at the same time among the set of features. This algorithm reduces the optimal bundling problem to a graph colouring problem. Firstly the algorithm constructs a graph with weighted edges, whose value is the number of conflicts between features. Then it sort these weights in the descending order. Due to the interference of noise, EFB algorithm solve the problem by tolerating skirmishes. Finally, it checks each feature to create bundles.

Besides, LightGBM adopts histogram-based algorithm (Lee & Goo, 2018) to speed up the training period.

3.3 The Proposed Model

In this section, we introduce how Deep Auto-Encoder based LightGBM model (DAEL) actually works. DAEL mainly consists of two stages. In the first stage, we learn expressive feature representation from a deep Auto-Encoder model that includes two hidden layers. Secondly, we use the intermediate result of the second hidden layer and the clean data as the input of the LightGBM model for final classification. The data is defined as: $D = \{(x_1, y_1), \dots, (x_m, y_m)\}, x \in R^m$. Figure 1. shows the architecture of the proposed model.

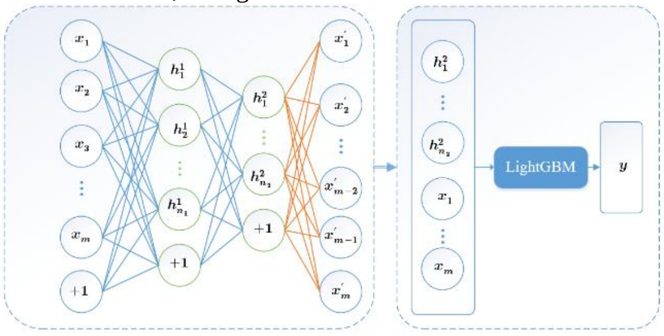

Figure 1: Deep Auto-Encoder based LightGBM architecture.

In the first stage, the deep Auto-Encoder consists of four layers: input layer, two hidden layers, output layer. Unlike other deep neural network, deep Auto-encoder is trained layer by layer. Firstly, the input layer gets input from clean data that includes 117 features. The first hidden layer contain 20 nodes, so we create the first simple Auto-Encoder model whose intermediate result is 20-dimensional vector. Then we use this intermediate result as the input of next hidden layer. In the second hidden layer, 20 features are compressed into 10 features. In this case, 10 features are presented the clean data in the whole deep Auto-encoder architecture.

In the second stage, there is a LightGBM model for the classification task. First of all, the intermediate result of the second layer and the clean dataset is concatenated into new dataset. Then, the new data are inputed into LightGBM model. Due to the imbalance of the data, macro-F1 (Yang, 1999) is adopted as the evaluation standard for training.

4 PERFORMANCE EVALUATION

4.1 Dataset

In this experiment, we evaluate our proposed model on use KDD 99 dataset which is the benchmark dataset in intrusion detection system. This dataset was derived from an intrusion detection assessment project at MIT Lincoln Laboratory (Sabhnani & Serpen, 2003). The dataset contains more than 5 million data, each of which contains 41 features and 1 label. We uses 494,021 samples for training the model and 311,029 samples for evaluating the model. It is common to use 10% of the origin data for training model. In the data, labels can be divided into two categories: Normal and attacks. More specifically, Attacks can be broken down into four categories. The detail of the dataset is as shown in the following Table 1.

Table 1: data distribution.

Label	Training set	Test set
Normal	97278	60593
DOS	391458	229853
R2L	1126	16189
U2R	52	228
Probe	4107	4166

4.2 Data Pre-processing

The KDD 99 dataset includes non-numerical features and duplicates, so this dataset is preprocessed before being inputted into models.

Firstly, the data are deduplicated and disambiguated. Because these duplicate data cause that the model assign a bigger weight to the more frequent data. We need to ensure that there is only one result for a piece of data, these data are disambiguated. After this work, the training dataset consists of 145585 samples and test dataset consists of 77291 samples.

Then, data transformation is applied to the experiment. The symbolic features (protocol_type, services and flag) are mapped to numeric feature by One-Hot Encoding (Buckman et al., 2018). For example, the feature ‘protocol_type’ contains three values: tcp, udp and icmp. These values are mapped to (1, 0, 0), (0, 1, 0) and (0, 0, 1) in turn. However, the label is mapped to numeric feature by Label-Encoding (Zhang et al., 2018). As we have seen in Table 1, the ‘label’ field contains five values, which are mapped to 0, 1, 2, 3 and 4 from top to bottom.

Since Auto-Encoder is used in the framework, it is necessary to standardize the data to eliminate differences caused by the different value scales between features. In this experiment, z-score method is adopted. The standard score of a raw score is calculated as

$$z = \frac{x-\mu}{\sigma} \tag{6}$$

where μ is the mean of population and σ is the standard deviation of the population.

4.3 Evaluation Metrics

We evaluate the performance of the proposed model based on the following metrics: Accuracy, macro-F1.

For the binary classification problems, it can be divided into the following four cases according to the real categories of data and the predicted results of the classifier:

- TP (True Positive): The positive class is predicted to be positive class.
- FP (False Positive): The negative class is predicted to be positive class.
- FN (False Negative): The positive class is predicted to be negative class.
- TN (True Negative): The negative class is predicted to be negative class.

Precision: Proportion of samples with positive correct predictions for all samples with positive predictions:

$$P = \frac{TP}{TP + FP} \times 100\% \quad (7)$$

Recall: Proportion of samples with positive correct predictions for all positive samples:

$$R = \frac{TP}{TP + FN} \times 100\% \quad (8)$$

F1-measure: Harmonic mean of Precision and Recall:

$$F1 = \frac{2 \times P \times R}{P + R} \quad (9)$$

In the multi-classification problem, it is assumed that there are k real categories, and the formula for calculating macro F1 rate is

$$F1_{Macro} = \frac{1}{n}\sum_{i=1}^{k} F1_i \quad (10)$$

5 EXPERIMENTAL RESULTS AND ANALYSIS

This experimental environment for the CPU: Razen1600X, GPU: GTX1070Ti, 16 g memory, operating system for Ubuntu. In order to prove the validity of LightGBM model, we compare it with some existing mainstream models. Table 2 shows the training time and accuracy for multi-classification.

Table 2: mainstream models comparison.

model	Training Time (s)	Acc (%)
LightGBM	422	94.7
XGBoost	4395	92.5
SVM	-	87.7
CNN	-	91.3
LR	79	86.3
RF	159	90.5

where '-' means these models are extremely slow to train.

According to the experimental results, the LightGBM algorithm has the best comprehensive performance. While ensuring high training efficiency, the accuracy of the model far exceeds the above other classifiers. Based on the characteristics of data imbalance, macro F1 is used as the evaluation standard in the training phase of this model. LightGBM training process F1 Score is shown in Figure 2:

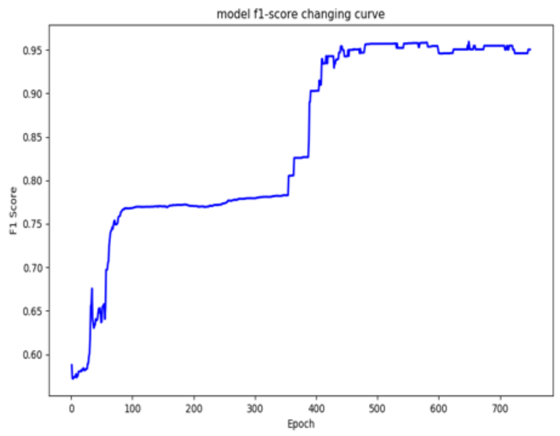

Figure 2: macro F1 changing curve.

As we have seen in Figure 2, LightGBM is hard to optimize after 500 epochs.

In order to prove the effectiveness of our DAEL model on IDS, we compared our algorithm with the previous implemented work. The Table 3 shows that our proposed model get better accuracy and Macro F1 score.

Tbale 3: model improvement.

Models	Acc (%)	Macro F1 (%)
LightGBM	94.7	95.6
DAE	94.2	93.5
DAEL	95.3	96.2

6 CONCLUSION

In this paper, we presented a deep Auto-Encoder based LightGBM approach for improving the intrusion detection system. A deep Auto-Encoder is used for digging some key features. Then LightGBM model is used to automatic feature selection and classify these data. The encoder idea is one of the most useful in the field deep learning and the LightGBM approach is widely used in various practical problems, so we can take full advantage of deep learning and traditional machine learning. KDD 99 dataset is used for evaluating the performance of our proposed model. The experimental result showed that our proposed method achieved accuracy 95.3% on the test dataset.

In future, we will further explore more deep learning methods to represent correlations between features and labels. Additionally, we will further analyse how to evaluate the performance of intrusion detection system more effectively.

REFERENCES

Aljawarneh, S., Aldwairi, M. & Yassein, M. B., 2018. 'Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model'. *Journal of Computational Science*. 25, 152-160.

Anderson, J. P., 1980. 'Computer security threat monitoring and surveillance'. *Technical Report*, James P. Anderson Company.

Bai, Y. & Kobayashi, H., 2003. 'Intrusion detection systems: technology and development'. In *International Conference on Advanced Information Networking & Applications*. IEEE, 710-715.

Buckman, J., et al., 2018. 'Thermometer encoding: One hot way to resist adversarial examples'. In *ICLR*.

Cheong, S., Oh, S. H. & Lee, S. Y., 2004. 'Support vector machines with binary tree architecture for multi-class classification'. *Neural Information Processing-Letters and Reviews*, 2(3), 47-51.

Farahnakian, F. & Heikkonen, J., 2018. 'A deep auto-encoder based approach for intrusion detection system'. In *20th International Conference on Advanced Communication Technology (ICACT)*. IEEE, 178-183.

Javaid, A., et al., 2016. 'A deep learning approach for network intrusion detection system'. In *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies*, 21-26.

Ke, G., et al., 2017. 'Lightgbm: A highly efficient gradient boosting decision tree'. In *Advances in Neural Information Processing Systems*. 3146-3154.

Kim, D. S. & Park, J. S., 2003. 'Network-based intrusion detection with support vector machines. International Conference on Information Networking'. In *International Conference on Information Networking*. Springer, Berlin, Heidelberg.747-756.

Lee, K. B. & Goo, H. W., 2018. 'Quantitative image quality and histogram-based evaluations of an iterative reconstruction algorithm at low-to-ultralow radiation dose levels: a phantom study in chest CT'. *Korean journal of radiology*, 19(1), 119-129.

Lee, W. & Stolfo, S. J., 2000. *A framework for constructing features and models for intrusion detection systems*. 3, 227-261.

Mahmood, H. A., 2018. 'Network Intrusion Detection System (NIDS) in Cloud Environment based on Hidden Naïve Bayes Multiclass Classifier'. *Al-Mustansiriyah Journal of Science*, 28(2), 134-142.

Manning, C. D. & Schütze, H., 1999. *Foundations of statistical natural language processing*, MIT press.

Nguyen, S. N., et al., 2018. 'Design and implementation of intrusion detection system using convolutional neural network for DoS detection. Proceedings of the 2nd International Conference on Machine Learning and Soft Computing'. In *Proceedings of the 2nd International Conference on Machine Learning and Soft Computing*. ACM, 34-38.

Puri, A. & Sharma, N., 2017. 'A novel technique for intrusion detection system for network security using hybrid svm-cart'. *International Journal of Engineering Development and Research*. Vol. 5.

Ribler, R. L., et al., 1998. 'Autopilot: Adaptive control of distributed applications'. In *IEEE International Symposium on High Performance Distributed Computing*. IEEE.

Sabhnani, M. & Serpen, G., 2003. 'Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context'. *MLMTA*, 209-215.

Umbaugh, S. E., 1997. C*omputer vision and image processing: a practical approach using cviptools with cdrom*, Prentice Hall PTR.

Yang, Y., 1999. 'An evaluation of statistical approaches to text categorization'. *Information retrieval*, 1(1-2), 69-90.

Zhang, Q., et al., 2018. *Category coding with neural network application*. arXiv preprint arXiv:1805.07927.