WTS 2018 1570420493

Autoencoder-based Network Anomaly Detection

Zhaomin Chen, Chai Kiat Yeo, Bu Sung Lee, Chiew Tong Lau
Computer Network and Communication Graduate Lab
School of Computer Science and Engineering
Nanyang Technological University, Singapore 639798
Email: {chen0954, asckyeo, ebslee, asctlau}@ntu.edu.sg

***Abstract*—Anomaly detection is critical given the raft of cyber attacks in the wireless communications these days. It is thus a challenging task to determine network anomaly more accurately. In this paper, we propose an Autoencoder-based network anomaly detection method. Autoencoder is able to capture the non-linear correlations between features so as to increase the detection accuracy. We also apply the Convolutional Autoencoder (CAE) here to perform the dimensionality reduction. As the Convolutional Autoencoder has a smaller number of parameters, it requires less training time compared to the conventional Autoencoder. By evaluating on NSL-KDD dataset, CAE-based network anomaly detection method outperforms other detection methods.**

***Index Terms*—Network Anomaly Detection, Autoencoder, Convolutional Autoencoder, Dimensionality Reduction, Reconstruction Error, NSL-KDD Dataset**

I. Introduction

Wireless communication has experienced an tremendous growth in network traffic in recent years. However, wireless networks are more vulnerable to the attacks compared to the traditional wired networks. As the sensor nodes in wireless communication lack of supervision, the malicious attacks can easily hack into the systems and waste many energy resources. Designing an efficient anomaly detection system is very essential to protect the wireless communication.

Dimensionality reduction based anomaly detection method is one of the popular detection methods. It is based on the assumption that the features of normal data are correlated with each other. Therefore, this method attempts to find a subspace which can optimally describe the normal data. Then, the data will be projected into that optimal subspace and those which have large reconstruction error will be determined as anomalies. Principal Component Analysis (PCA) based method is a category of this kind of anomaly detection. Recent works[9], [13], [17] have revealed its effectiveness in the area of anomaly detection.

However, PCA is a linear transformation which fails to capture the non-linear correlations between features. In network anomaly detection, most of the features are non-linearly correlated. Therefore, PCA-based anomaly detection may cause many false alarms. Autoencoder is a novel dimensionality reduction method [8]. By using neural networks, it can find the optimal subspace which captures the non-linearly correlations between features. In this paper, we introduce an Autoencoder-based anomaly detection system which employs Autoencoder to perform dimensionality reduction. In addition, in order to reduce the number of parameters in Autoencoder, we also propose a Convolutional Autoencoder (CAE) based anomaly detection method. As CAE can share the parameters in convolutional and deconvolution layers, it has fewer parameters than the conventional Autoencoder. As far as we know, this is the first time CAE is being applied to the anomaly detection.

The main contributions of our work are as follows. Firstly, we propose the Autoencoder-based and CAE-based network anomaly detection methods. Secondly, by evaluating our methods using the NSL-KDD dataset, we compare the detection performance among conventional autoencoder, CAE, PCA and other detection methods. The evaluation results show that CAE-based detection method outperforms other detection methods.

In the following, we discuss the related work on network anomaly detection firstly. In Section III, we provide the methodology of Autoencoder-based and CAE-based network anomaly detection systems. In Section IV, we present the evaluation results. In Section V, we conclude the whole paper.

II. Related Works

Anomaly detection method has received many attentions over the past years. It can be generally categorized into these types: statistical methods, neighbor-based methods and dimensionality reduction based methods. Statistical methods [5], [6] attempt to fit the distributions on the training data. Then any data which has low probability under this distribution will be determined as an anomaly. Neighbor-based methods [7], [12] assume that normal data has relatively more neighbors than the anomalous data. Dimensionality reduction based methods utilize the reconstruction error to classify the anomalies. PCA is a well known dimensionality reduction method. Therefore it has been applied to anomaly detection in recent papers [1], [2], [3], [4], [17].

However, as PCA can only capture the linear correlation between features, it may cause many false alarms. Autoencoder has been proven to be a better dimensionality reduction methods than PCA [8]. It is able to reveal the non-linear correlations among features. Therefore, in this paper, we introduce an Autoencoder-based network anomaly detection method. In addition, we also apply the Convolutional Autoencoder to the network anomaly detection. As Convolutional Autoencoder has fewer parameters than the conventional Autoencoders, it requires much less time to train.

III. Methodology of Autoencoder-based Anomaly Detection

A. Dimensionality Reduction based Anomaly Detection

Dimensionality Reduction based Anomaly Detection method attempts to find an optimal subspace where the normal data and anomalous data appear very different.

Let us assume that the normal training set is $\{x_1, x_2, ..., x_n\}$, each of which is a d dimensional vector ($x_i \in R^d$). In the training phase, we construct a model to project these training data into the lower dimensional subspace and reproduce the data to get the output $\{x'_1, x'_2, ..., x'_n\}$. Therefore, we optimize the model to minimize reconstruction error so as to get the optimal subspace. The reconstruction error is defined as:

$$\varepsilon(x_i, x'_i) = \sum_{j=1}^{d} (x_i - x'_i)^2 \tag{1}$$

One of the famous dimensionality reduction model is Principal Component Analysis (PCA). In PCA, by selecting the first k dominant Principal Components, we can get the optimal subspace $V_k \in R^{m \times k}$. Therefore, we project the test data into this subspace and reconstruct the data. The error in Eq. 1 can be further calculated as:

$$\varepsilon(x_i, x_i{}') = \left\| x_i (I - V_k V_k^T \right\|^2 \tag{2}$$

As the normal data in the test dataset meet the normal profile which is built in the training phase, the corresponding error is smaller, whereas the anomalous data will have a relatively higher reconstruction error. As a result, by thresholding the reconstruction error, we can easily classify the anomalous data:

$$c(x_i) = \begin{cases} normal & \varepsilon_i < \theta \\ anomalous & \varepsilon_i > \theta \end{cases} \tag{3}$$

In this paper, we implement PCA-based anomaly detection method to compare with Autoencoder-based Anomaly Detection methods.

B. Conventional Autoencoder

Here we briefly describe the architecture of the conventional Autoencoder. Normally, Autoencoder consists of the *encoder* and the *decoder*. Fig. 1 presents a basic Autoencoder.

Encoder: In Fig. 1, the input vectors ($x_i \in R^d$) are compressed into $m (m < d)$ numbers of neurons which make up the hidden layer. The activation of the neuron i in the hidden layer is given by:

$$h_i = f_\theta(x) = s(\sum_{j=1}^{n} W_{ij}^{input} x_j + b_i^{input}) \tag{4}$$

where x is the input vector, θ is the parameters $\{W^{input}, b^{input}\}$, W represents the encoder weight matrix with size $m \times d$ and b is a bias vector of dimensionality m. Therefore, the input vector is encoded to a lower-dimensional vector.

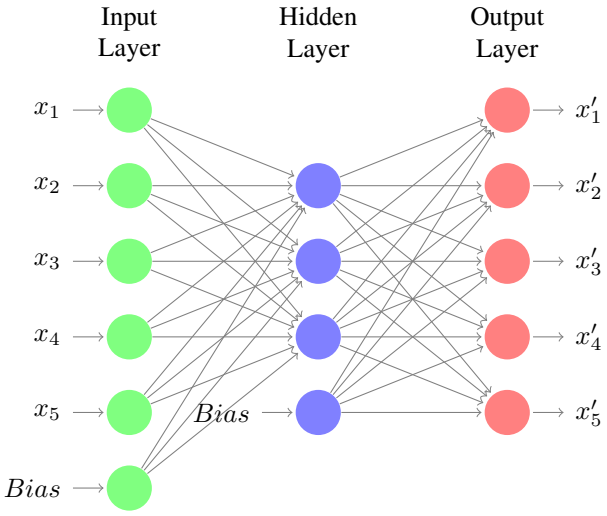


Fig. 1: Autoencoder Architecture

Decoder: The resulting hidden representation h_i is then decoded back to the original input space R^d. The mapping function is as follows:

$$x'_i = g_{\theta'}(h) = s(\sum_{j=1}^{n} W_{ij}^{hidden} h_j + b_i^{hidden}) \tag{5}$$

The parameter set of the decoder is $\theta' = \{W^{hidden}, b^{hidden}\}$.

We optimize the Autoencoder to minimize the average construction error with respect to θ and θ':

$$\begin{aligned} \theta^*, \theta'^* &= \arg\min_{\theta,\theta'} \frac{1}{n} \sum_{i=1}^{n} \varepsilon(x_i, x_i{}') \\ &= \arg\min_{\theta,\theta'} \frac{1}{n} \sum_{i=1}^{n} \varepsilon(x_i, g_{\theta'}(f_\theta(x_i))) \end{aligned} \tag{6}$$

where ε is the reconstruction error defined in Eq. 1. After we finish training this Autoencoder, we can feed the test data into it to compute the reconstruction errors for each set of data. The anomalous data can be determined by utilizing Eq. 3.

To be noted here, the activation functions f and g should be non-linear functions so as to reveal the non-linear correlation between the input features. In this paper, we select sigmoid function as the activation function.

C. Convolutional Autoencoder

Convolutional Autoencoder (CAE) is a special kind of Autoencoder which does not apply the fully connected neural layer. The CAE model consists of convolutional and deconvolution layers. It utilizes convolutional layer in the encoder part and deconvolutional layer in the decoder part. To be noted here, the input of CAE should be a 2D data.

The convolutional/deconvolutional layer can be expressed by the following equation:

$$h_{l+1}^k = \sigma(\sum_{j \in J} x_l^j \otimes w_l^k + b^k) \tag{7}$$

where h_{l+1}^k is the latent representation of $k-th$ feature map in layer $l+1$, σ is a non-linear activation function (we also select sigmoid function for CAE), x_l^j is the $j-th$ feature map of the output in layer l, w_l^k is the $k-th$ filter weight for the layer l and b^k is the bias parameter. In this equation, $\otimes$ represents the 2D convolution operation. If x_l^j has the size of $W \times W$ and the size of filter is $F \times F$, the output of a convolutional layer will have the size of $(W-F+1)\times(W-F+1)$ assuming that we utilize stride 1 and no zero-padding. For a deconvolution layer, the size of the output will be $(W+F-1)\times(W+F-1)$.

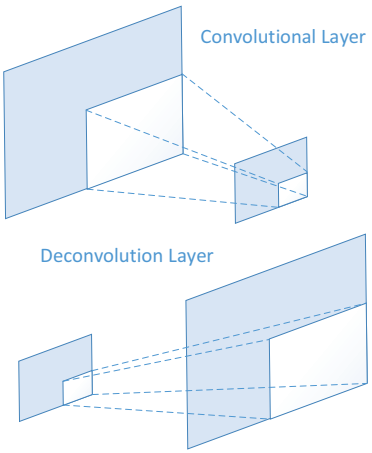


Fig. 2: Illustration of Convolutional and Deconvolution Layers

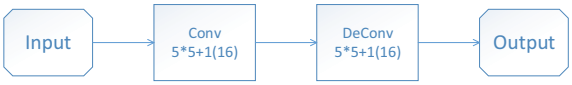


Fig. 3: Architecture of the CAE

Fig. 2 shows an illustration of the convolutional and deconvolution layers. Apparently, convolutional layer is able to decrease the feature number while the deconvolution layer can increase the feature number. As a result, in CAE, the convolutional layer takes the role of encoder to perform the dimensionality reduction, while the deconvolution layer is applied here to reconstruct the data. Similar to the traditional Autoencoder, we train the CAE network to minimize the average reconstruction error.

As presented in Fig. 3, the exact structure of CAE utilized in this paper is as follows:

1) A 5×5 convolutional layer with 16 filters to encode the original data into the hidden latent representation with 16 feature maps
2) A 5×5 deconvolution layer with 16 filters to decode the hidden representation

CAE takes advantage of the Convolutional and Deconvolution layers. As these layers utilize the kernel filters, parameter sharing scheme is applied here to control the number of parameters. Therefore, compared with conventional Autoencoder, CAE has smaller number of parameter so that the training time of CAE is much smaller.

D. Framework of the Autoencoder-based Network Anomaly Detection

The Framework of Autoencoder-based Network Anomaly Detection System consists of the following steps:

Step 1: Data Preprocessing In this step, we need to normalize the data so as to pre-scale all the features into a specific interval. In our experiments, we apply the unity-based normalization by using Eq.8. Therefore, we can convert all the features into the range of $[0, 1]$.

$$X' = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{8}$$

Step 2 (Only for CAE): 2D Data Generation This step is only for the Convolutional Autoencoder, as CAE can only be applied to the 2D data. By applying the Triangle Area Maps (TAMs) method in [14]. we can easily get the 2D data. The detail steps of TAM method are as follows: For a d-dimensional data $\vec{x} = (x_1, x_2, ..., x_d)$, we construct the 2D data based on the following equation: $y_{ij} = x_i \times x_j, i \in [1 : d], j \in [1 : d]$. Therefore, we can get a $d \times d$ matrix for each input 1D vector.

Step 3: Autoencoder Training In the training phase, we just utilize the normal data to train the conventional Autoencoder / CAE. We optimize the Autoencoder to minimize the average reconstruction error. Therefore, by training the Autoencoder, we can build the normal profiles of the legitimate traffic data.

Step 4: Autoencoder Testing In this step, we feed the test dataset into the Autoencoder model. We also utilize the reconstruction error as the anomalous scores. Therefore, any data whose score is larger than the threshold will be determined as an anomaly.

IV. Evaluation

Currently, there are only a few public available network intrusion datasets, among which KDD Cup 99 Dataset is the most famous one. This dataset has been widely used in the literature. However, in [15], the authors addressed two important issues of KDD Cup 99 dataset, which could highly degrade the detection accuracy of the detection systems. In order to address these issues, they provided a new dataset: NSL-KDD Dataset, which is composed of selected data from the complete KDD 99 dataset. As a result, in this paper, we evaluate our detection systems based on NSL-KDD dataset.

In order to evaluate our detection systems, the following metrics are applied here:

- True Positive Rate (TPR) TP/(TP+FN)
- False Positive Rate (FPR) FP/(FP+TN)
- Detection Accuracy (TP+FN)/(TP+TN+FN+FP)

where TP, FP, TN, FN are short for True Positive, False Positive, True Negative, False Negative respectively. In addition, by thresholding the reconstruction errors, we can get the Receiver Operating Characteristic (ROC) curve for each detection system. AUC score will be adopted to compare the detection performances. AUC score is the area undet the ROC

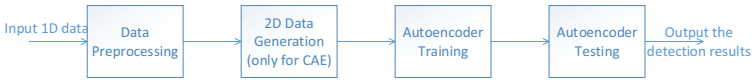
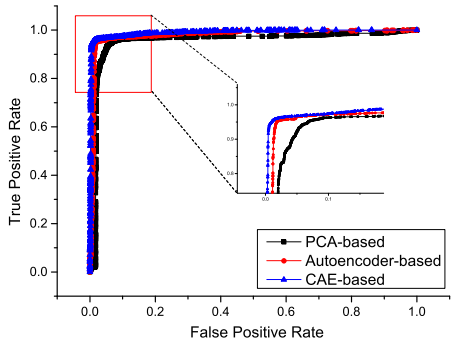
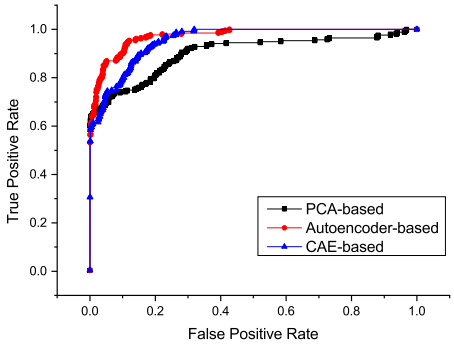


Fig. 4: Framework of the Autoencoder-based Network Anomaly Detection

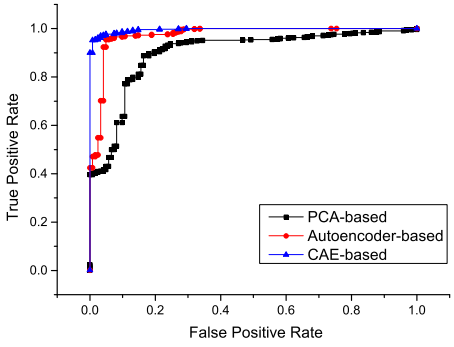
curve. Therefore, the larger the AUC score is, the better is the detection performance.



(a) ROC curves for Detecting TCP Traffic Data



(b) ROC curves for Detecting UDP Traffic Data



(c) ROC curves for Detecting ICMP Traffic Data

Fig. 5: ROC Curves of three Detection Systems

TABLE I: AUC scores of the different detection systems

	PCA	Autoencoder	CAE
TCP	0.9360	0.98724	**0.9902**
UDP	0.8937	**0.9658**	0.9564
ICMP	0.8941	0.9822	**0.9869**
All	0.9292	0.9857	**0.9884**

A. Comparing with PCA-based Anomaly Detection

In this subsection, we implement three different anomaly detection systems: PCA-based, Conventional Autoencoder based and CAE-based detection systems. For PCA-based system, we select 16 dominant Principal Components empirically. For the Conventional Autoencoder based system, we set the hidden layer neuron number as 8. For CAE-based detection system, we adopt the architecture of Fig. 3.

We split the whole NSL-KDD Dataset according to the network traffic types (TCP, UDP and ICMP) and conduct a tenfold cross-validation for each traffic type. By thresholding the reconstruction error in Eq. 1, we can derive TPR, FPR and AUC score of each detection system.

Fig. 5 plots the ROC curves of three detection systems for detecting TCP, UDP and ICMP traffic data. Table I presents the AUC scores of three anomaly detection systems for different traffic types. As we can see, the AUC scores of Conventional Autoencoder and CAE are all better than that of PCA. For TCP and ICMP data, CAE-based anomaly detection system achieves the largest AUC scores. For UDP data, Conventional Autoencoder based anomaly detection system performs the best. As the Conventional Autoencder and CAE both select the non-linear functions to activate the neurons, they can extract the non-linear correlation between multiple features. However, PCA is just a linear transformation which is only able to capture the linear correlation. As a result, Autoencoders can successfully detect more attacks than PCA.

For CAE, we need to utilize TAM method to convert the original input into the 2D data. This also provides more statistical information which could increase the detection accuracy. This is the main reason that CAE performs better than the conventional Autoencoder. In addition, as CAE utilizes the convolutional layer and deconvolution layer to encode and decode the data, CAE has fewer parameters than the conventional Autoencoder, which makes CAE converge much faster.

B. Comparing with other Anomaly Detection Systems

In this subsection, we present a general comparison between our two autoencoder based anomaly detection systems and other famous detection systems: k-Nearest Neighbor (k-NN) [11], Support Vector Machine (SVM) [10] and Triangle area

TABLE II: Comparisons between Different Network Anomaly Detection Systems

	False Positive Rate(%)	Detection Accuracy(%)
k-NN	38.02	88.91
SVM	6.91	92.98
TANN	3.83	96.91
AE	4.09	95.85
CAE	3.44	96.87

based Nearest Neighbors (TANN) [16]. The detection results of these three detection systems are quoted from [16].

Here, we compare these systems in terms of FPR and Detection Accuracy. The comparison results are presented in Table II. As we can see from the table, Autoencoder and CAE perform much better than k-NN and SVM. That is, they provide higher Detection Accuracy and lower False Positive Rate. When compared with TANN, the conventional Autoencoder performs slightly worse while CAE has similar detection performance. However, compared with TANN, Autoencoder and CAE are much easier for future retraining. We can easily retrain them according to the new incoming data.

V. Conclusion

In this paper, we propose two Autoencoder-based network anomaly detection systems. By applying Autoencoders to perform dimensionality reduction, we can easily capture the non-linear correlations between features. In addition to the conventional Autoencoder, we also apply the Convolutional Autoencoder here so as to reduce the training time. The evaluation results show that the Convolutional Autoencoder based anomaly detection method outperforms other detection methods.

References

[1] Z. Chen, C. K. Yeo, B. S. L. Francis, and C. T. Lau. A mspca based intrusion detection algorithm tor detection of ddos attack. In *Communications in China (ICCC), 2015 IEEE/CIC International Conference on*, pages 1–5. IEEE, 2015.

[2] Z. Chen, C. K. Yeo, B. S. L. Francis, and C. T. Lau. Combining mic feature selection and feature-based mspca for network traffic anomaly detection. In *Digital Information Processing, Data Mining, and Wireless Communications (DIPDMWC), 2016 Third International Conference on*, pages 176–181. IEEE, 2016.

[3] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau. Detection of network anomalies using improved-mspca with sketches. *Computers & Security*, 65:314–328, 2017.

[4] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau. A novel anomaly detection system using feature-based mspca with sketch. In *Wireless and Optical Communication Conference (WOCC), 2017 26th*, pages 1–6. IEEE, 2017.

[5] A. Dukkipati, D. Ghoshdastidar, and J. Krishnan. Mixture modeling with compact support distributions for unsupervised learning. In *Neural Networks (IJCNN), 2016 International Joint Conference on*, pages 2706–2713. IEEE, 2016.

[6] E. Eskin. Anomaly detection over noisy data using learned probability distributions. In *In Proceedings of the International Conference on Machine Learning*. Citeseer, 2000.

[7] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo. A geometric framework for unsupervised anomaly detection. In *Applications of data mining in computer security*, pages 77–101. Springer, 2002.

[8] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[9] A. Lakhina, M. Crovella, and C. Diot. Diagnosing network-wide traffic anomalies. In *ACM SIGCOMM Computer Communication Review*, volume 34, pages 219–230. ACM, 2004.

[10] K.-L. Li, H.-K. Huang, S.-F. Tian, and W. Xu. Improving one-class svm for anomaly detection. In *Machine Learning and Cybernetics, 2003 International Conference on*, volume 5, pages 3077–3081. IEEE, 2003.

[11] Y. Liao and V. R. Vemuri. Use of k-nearest neighbor classifier for intrusion detection. *Computers & security*, 21(5):439–448, 2002.

[12] S. S. Rakhe and A. S. Vaidya. Enhanced outlier detection for high dimensional data using different neighbor metrics. *International Journal of Engineering Science*, 1568, 2016.

[13] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang. A novel anomaly detection scheme based on principal component classifier. Technical report, DTIC Document, 2003.

[14] Z. Tan, A. Jamdagni, X. He, P. Nanda, and R. P. Liu. A system for denial-of-service attack detection based on multivariate correlation analysis. *IEEE transactions on parallel and distributed systems*, 25(2):447–456, 2014.

[15] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani. A detailed analysis of the kdd cup 99 data set. In *Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium on*, pages 1–6. IEEE, 2009.

[16] C.-F. Tsai and C.-Y. Lin. A triangle area based nearest neighbors approach to intrusion detection. *Pattern recognition*, 43(1):222–229, 2010.

[17] H. Xu, C. Caramanis, and S. Mannor. Outlier-robust pca: the high-dimensional case. *IEEE transactions on information theory*, 59(1):546–572, 2013.