Journal of Computer Science and Cybernetics, V.36, N.2 (2020), 159–172
DOI 10.15625/1813-9663/36/2/14578

A DOUBLE-SHRINK AUTOENCODER FOR NETWORK ANOMALY DETECTION

CONG THANH BUI[1], LOI CAO VAN[2], MINH HOANG[3], QUANG UY NGUYEN[2,*]

[1]*Software R&D Department, Hi-tech Telecommunication Center, Communications Command*
[2]*Faculty of Information Technology, Le Quy Don Technical University*
[3]*Head Office, Institute of Science Technology and Innovation*

Abstract. The rapid development of the Internet and the wide spread of its applications has affected many aspects of our life. However, this development also makes the cyberspace more vulnerable to various attacks. Thus, detecting and preventing these attacks are crucial for the next development of the Internet and its services. Recently, machine learning methods have been widely adopted in detecting network attacks. Among many machine learning methods, AutoEncoders (AEs) are known as the state-of-the-art techniques for network anomaly detection. Although, AEs have been successfully applied to detect many types of attacks, it is often unable to detect some difficult attacks that attempt to mimic the normal network traffic. In order to handle this issue, we propose a new model based on AutoEncoder called Double-Shrink AutoEncoder (DSAE). DSAE put more shrinkage on the normal data in the middle hidden layer. This helps to pull out some anomalies that are very similar to normal data. DSAE are evaluated on six well-known network attacks datasets. The experimental results show that our model performs competitively to the state-of-the-art model, and often out-performs this model on the attacks group that is difficult for the previous methods.

Keywords. Deep learning; AutoEncoders; Anomaly detection; Latent representation.

1. INTRODUCTION

Due to the fast increase of many cyber-attacks, seeking for solutions to identify and prevent these attacks is a major task in network security. Recently, anomaly detection techniques have been widely applied for automatically identifying intruders, malware and malicious activities in network systems [13, 3]. Typical security systems for carrying out this task are known as Intrusion Detection Systems (IDSs). IDSs can be used as the second security layer behind firewalls to protect computer networks from illegal and malicious activities [13, 23]. In IDSs, the anomaly detection-based approach is getting more popular because of its ability in detecting new network attacks. Among several techniques for anomaly detection, machine learning techniques have been used as the main approach [12, 17, 2]. However, machine learning-based anomaly detection still suffers from some obstacles such as high-dimensional data, the continuous evolution of network attacks, and the lack of attack data and its labels for training the detection models [14, 11, 31].

Machine learning algorithms for anomaly detection are often divided into three categories: supervised, unsupervised and semi-supervised learning. Of these, one-class classification (OCC) can eliminate the challenges of lacking attack data and its corresponding labels since they can construct anomaly detection models from only one class of data, usually the normal data. Any data point that

*Corresponding author.
E-mail addresses: congthanhttmt@gmail.com (C.T.Bui); loicv79@gmail.com (L.C.Van); hoangminh@most.gov.vn (M.Hoang); quanguyhn@gmail.com (Q.U.Nguyen).

does not fit the models well will be classified as anomaly [26]. OCCs have been used for identifying anomalies in network security in many researches [10, 9, 33, 11]. However, OCC-based methods, such as Support Vector Machine (SVM) [30] and Local Outlier Factor (LOF) [5], often perform inefficiently when the data is high-dimensional. Moreover, it is difficult to turn hyper-parameters for these methods [28, 11].

Recently, Autoencoders (AEs) have been widely used for anomaly detection and they achieved the state-of-the-art result in many researches [20, 18, 29]. AEs are feedforward neural networks which learn to reconstruct the input at the output layer [4, 19]. The middle hidden layer attempts to compress data into a lower-dimensional feature space: preserving non-redundant information while removing the redundancies of the input data. A trained AE on normal data will behave well on normal examples yielding small reconstruction errors (REs), whereas this AE will poorly reproduce on anomaly instances creating larger REs. Thus, REs can be used as anomaly scores. Alternatively, the middle hidden layer of an AE can be used as a new feature representation (latent representation) and this latent presentation is the input to other anomaly detection algorithms [11, 9, 27, 6, 8].

Shrink AE (SAE) [11] is a typical model for learning the feature representation. SAE is an extension of an ordinary AE by incorporating a regularizer to the loss function of the AE. SAE learn to reconstruct the normal data as well as force this data into a tiny region in the latent space. SAE achieves convincing results on a wide range of anomaly detection datasets. However, this model did not perform well on some types of attacks, such as the Remote to Local attack group (R2L) [11, see Tab. 3], which represent the most dangerous attack types [24]. The reason could be that R2L embeds itself in data packets, and does not create a sequential patterns like DoS and Probe attacks, thus the R2L traffic is very similar to the normal one [24, 1, 21]. In this paper, we propose a novel model based on AE that attempts to address the shortcomings of SAE. The main contributions of this paper are:

- We propose a new model namely Double-Shrink AutoEncoder (DSAE) for detecting some particular types of network attacks which are difficult for SAE.
- We conduct extensive experiments on the well-known datasets, both synthetic and real datasets. The results suggest that DSAE is better than SAE in detecting critical attacks such as R2L.

The remaining parts of the paper are as follows. In Sections 2 and 3, we briefly present some related work, and a complementary background of the paper. The proposed model, DSAE, is presented in Section 4, Sections 5 and 6 describe our experimental settings and analyze experimental results. Finally, we draw some conclusions and suggest the future work in Section 7.

2. BACKGROUND

This section briefly presents anomaly detection methods including Centroid, Autoencoder and Shrink Autoencoder. These techniques will be used for developing and evaluating our proposed methods in the following sections.

2.1. Centroid (CEN)

Centroid (CEN) [11] is a parametric method which uses a single Gaussian to model the training data. Let $X = \{x_1, x_2, ..., x_n\} \in \mathbb{R}^d$ be the training data, μ_j and σ_j be the mean and standard

deviation of the j-th feature of the training data respectively, n is the number of data samples, X is normalized by the z-score transformation on each feature using Equation (1)

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}, \tag{1}$$

where x_{ij} is the j-th element of x_i, and z_{ij} is its z-score. In testing process, the distance from the tested sample to the centroid is calculated and it reflects the anomaly degree of the observation.

2.2. AutoEncoder

An AutoEncoder (AE) is a type of artificial neural network [4, 19] that consists of two parts: an encoder and a decoder. The encoder attempts to map input data into a latent feature space in the hidden layer. This latent space is also called a bottleneck. Let f_θ be the encoder, and $X = \{x_1, x_2, ..., x_n\}$ be a dataset. The encoder f_θ will map the input $x_i \subseteq X$ into a latent vector $z_i = f_\theta(x_i)$ in the bottleneck. The decoder g_θ learns to reconstruct the input data at the output layer $\hat{x}_i = g_\theta(z_i)$ from the latent representation z_i. The encoder and decoder are usually represented in a form of an affine mappings as follows $f_\theta(x) = s_f(Wx + b)$ and $g_\theta(z) = s_g(W'z + b')$, where W, $W^{'}$ are the weight, b and $b^{'}$ are the bias, and s_f and s_g are the activation functions of the encoder and decoder. The AE is trained to minimize the loss function ($Loss_{AE}(\theta)$) called the reconstruction error (RE). RE is the difference between the input x_i and its reconstruction $\hat{x}_i$ and it can be measured by mean square error (MSE) for real-valued data, and cross entropy for binary data. When using MSE, the loss function of an AE over training samples can be calculated as Equation 2.

$$Loss_{AE}(\theta) = \frac{1}{m}\sum_{i=1}^{m}(x_i - \hat{x}_i)^2, \tag{2}$$

where θ represents the parameters of the AE, m is the number of training examples.

2.3. Shrink Autoencoder

Shrink Autoencoder (SAE) is an extension of AEs introduced recently by Cao et al. [11]. In SAE, a new regularizer is added to the loss function of an AE. This aims to encourage the AE to construct a compact feature representation for normal data in the hidden layer which facilitates the performance of the anomaly detection algorithms that are applied on the top of the feature representation. The AE loss function in Equation (2) can be redefined as follows

$$Loss_{SAE}(\theta) = Loss_{RE}(\theta) + Regularizer(\theta). \tag{3}$$

The first component in Equation (3) is RE (the loss of an ordinary AE), and the second one is the regularized term which is designed to restrict the hidden data to be centered at the origin in the latent feature space. The SAE loss can be precisely defined as

$$Loss_{SAE}(\theta) = \frac{1}{m}\left(\sum_{i=1}^{m}(x_i - \hat{x}_i)^2 + \alpha\sum_{i=1}^{m}\|z_i\|^2\right), \tag{4}$$

where $\hat{x}_i$ and z_i are the reconstruction and the latent vector of the observation x_i, respectively. m is the number of training examples, and α is the trade-off parameter.

3. RELATED WORK

This section presents a brief review of the previous research on anomaly detection using AutoEncoders. AutoEncoders can be employed for anomaly detection following two common approaches: (1) Using AEs stand-alone [18, 15]; (2) Using the bottleneck layer as a new feature representation in hybrid models [14, 11, 9, 28, 27, 35].

In the first approach, an AE can be trained on only normal data to minimize the reconstruction error, $\|x - \hat{x}\|^2$. The trained AE will captures the normal behavior well, and it yields small REs on normal examples. Anomalies that are often different from normal data will not fit the model and have large REs. Thus, REs are usually used as the anomaly score of a querying example. Hawkins et al. [18] used the RE of an AE with narrow middle layer as an indicator of anomalies. The model was evaluated on the Wisconsin Breast Cancer and the KDD99 datasets. The experimental results showed that the model produced very high accuracy. Fiore et al. [15] constructed an AE architecture, called Discriminative Restricted Boltzmann Machines (DRBM), to test the hypothesis that normal behaviors may exist a deep similarity. They expected that their model can represent all the common characteristics of normal traffic, and use them to distinguish unseen anomalous traffic from normal traffic. The authors evaluated their model on both real-world network traces and the KDD99 datasets. Their experimental results confirmed that their model could perform efficiently on data collected from the same network as that in the training phase. When the testing environment being greatly different from the training one their model produced poor performance.

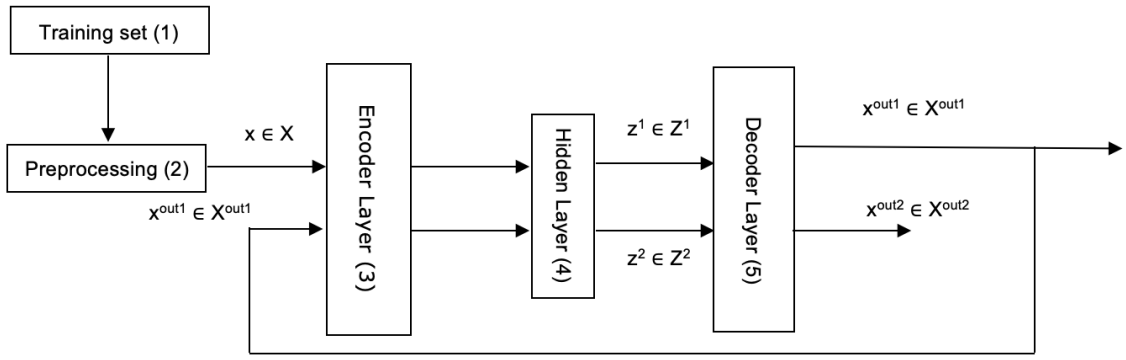
In some scenarios, well-known anomaly detection algorithms often suffer challenges posed by data such as the curse of dimensionality, irrelevant/redundant features. Therefore, the bottleneck layer of AEs have also been used as a new feature representation of lower dimension. Rajashekar et al. [27] proposed a hybrid between an AE and a self-organizing map (SOM) for representing the smartphone users' normal behavior. In this model, the authors used an AE to capture the most important characteristics of the users behavior. A SOM was then stacked on the output of the encoder of the AE. Veeramachaneni et al. [35] introduced an ensemble model that combines three single anomaly detection models: AE-based, density-based, and matrix decomposition-based models. The models were evaluated on a large network log dataset, and produced a good performance. Erfani et al. [14] used an AE architecture, called deep belief network (DBN), to enhance the performance of anomaly detection techniques in solving the curse of dimensionality in high-dimensional data. OCSVMs were then built on top of the DBN. This model has the advantages of the nonlinear feature reduction of the DBN and high detection accuracy of OCSVMs.

Recently, Cao et al. [11] proposed a new version of AE called Shrink AutoEncoder (SAE) for anomaly detection. Their model forces the hidden representation of normal data into a very tight area centered at the origin of the bottleneck by adding a new regularizer to the loss function of the AE. The hidden layer of the trained SAE will be then used as a new data representation for well-known anomaly detection algorithms, such as OCSVM. The experimental results showed that their approaches achieved very convincing results compared to the state-of-the-art AEs on a wide range of problems.

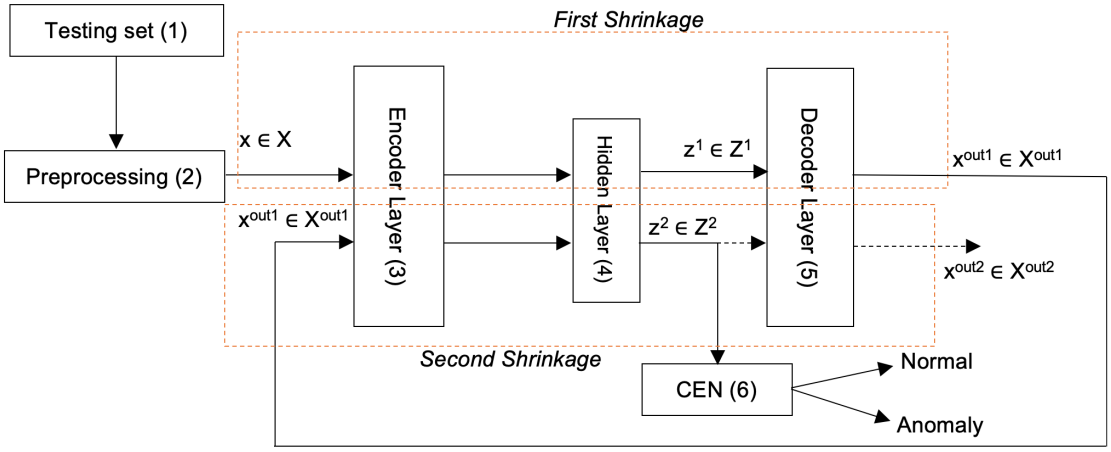
In this paper, we will propose a new model to leverage the performance of SAE in identifying attacks that are difficult for SAE. The detailed description of our model will be described in the next section.

4. DOUBLE-SHRINK AUTOENCODER

Our model attempts to detect the attacks that are difficult for SAE [11]. As mentioned in the privious section, SAE works based on the way, the encoder will lead normal data toward the origin, whereas anomalies will push far away from the center.



(a) Training model



(b) Testing model

Figure 1. Our proposal model

We assume that SAE is failed to detect some kinds of attacks that mimic the normal data. In this scenario, the latent vectors of these attacks are close to the latent vectors of the normal region in SAE. Our model aims to increase the shrink pressure on the normal latent vectors to separate these attacks from the normal vectors. In other words, we incorporate one more shrink pressure into SAE to form a new model called Double-shrink AutoEncoder (DSAE). Figure 1 presents the training and testing phases of DSAE. DSAE is trained by only normal data, the normal data is passed as the first input to DSAE. The output of the decoder will then also be used as the second input of DSAE. The model learns to minimize the reconstruction losses and driving latent vectors (z^1 and z^2) towards the center of the latent space. So that, the attack network traffic that are very similar to normal network traffic after pass the first shrinkage will also produce a small reconstruction errors (REs), but indeed, that output vector should be more likely to anomalous so by pushing to the second shrinkage, it will

lead the latent vector (z^2) go far away. In the training phase, the normal data is passed as the first input to DSAE. The output of the decoder will then also be used as the second input of DSAE. The model learns to minimize the reconstruction losses (call RE_1 and RE_2 and driving latent vectors (z^1 and z^2) towards the center of the latent space, $z^1 = f_\theta(x)$, that is latent vector at the bottleneck of the first shrink, $z^2 = f_\theta\left(g_\theta\left(z^1\right)\right)$ is latent vector at the bottleneck of the second shrink. In the testing phase, z^2 is used as the new feature representation for subsequent anomaly detection methods such as CEN.

The first and the second reconstruction errors (RE_1 and RE_2) can be defined as follows

$$L_{RE_1}(\theta, x_i) = \frac{1}{m}\sum_{i=1}^{m}\left(x_i - x_i^{out1}\right)^2, \tag{5}$$

$$L_{RE_2}(\theta, x_i) = \frac{1}{m}\sum_{i=1}^{m}\left(x_i^{out1} - x_i^{out2}\right)^2, \tag{6}$$

where θ is the parameters of DSAE, m is the number of the training data points, and x_i, x_i^{out1} and x_i^{out2} are the input data point i and its reconstruction values in the first and second shrinks respectively. Thus, the RE component of the DSAE loss can be written as in Equation (7)

$$L_{RE}(\theta, x_i) = L_{RE_1}(\theta, x_i) + L_{RE_2}(\theta, x_i). \tag{7}$$

The shrink term of the DSAE loss is the sum of the first shrink and the second shrink components in the following equations.

$$L_{Z_1}(\theta, x_i) = \frac{1}{m}\sum_{i=1}^{m}\left\|z_i^1\right\|^2, \tag{8}$$

$$L_{Z_2}(\theta, x_i) = \frac{1}{m}\sum_{i=1}^{m}\left\|z_i^2\right\|^2, \tag{9}$$

$$L_Z(\theta, x_i) = L_{Z_1}(\theta, x_i) + L_{Z_2}(\theta, x_i), \tag{10}$$

where z_i^1 and z_i^2 are the latent vectors of the input data x_i under the first shrink and the second shrink pressures respectively, and m is the number of the training data points.

The last component of the DSAE lost function is a weight decay regularizer on the network parameters W presented in Equation (11) below

$$L_W(\theta, x_i) = \sum_{l=1}^{L}\left\|W^l\right\|_F^2, \tag{11}$$

where $\|.\|_F$ denotes the Frobenius norm [17].

Therefore, the loss function of DSAE is defined as follows

$$L_{DSAE}(\theta, x_i) = L_{RE}(\theta, x_i) + \alpha * L_Z(\theta, x_i) + \beta * L_W(\theta, x_i), \tag{12}$$

where α and β are parameters for controlling the trade-off between the three terms in the DSAE loss function. In this paper, we naturally choose $\alpha = 10$ followed the settings by Cao et al. [11], and $\beta = 0.001$ as in [34].

5. EXPERIMENTAL SETTINGS

This section describes the datasets and the experimental settings used in the paper.

5.1. Datasets

Our experiments aims to highlight the performance of DSAE in detecting the attacks that are difficult for SAE. We employed three well-known network datasets [7] for our experiments, namely CTU13 [16], UNSW-NB15 [25] and NSL-KDD [32]. In these datasets, we consider all kinds of attacks as anomalies while the data created by users is normal.

Table 1. Datasets using for experiments

Dataset	**Dimension origin/after one-hot encoding**	**Trainning set**	**Testing sets**	
			Normal	***Abnormal***
NSLKDD	44/122	67343	9711	12833
UNSW-NB15	47/196	56000	37000	45332
CTU-13_08	16/40	29128	43694	3677
CTU-13_09	16/41	11986	17981	110998
CTU-13_10	16/38	6338	9509	63812
CTU-13_13	16/40	12775	19164	24002

- NSL-KDD [32] is an effective benchmark for anomaly detection methods that consists of a training set (KDDTrain+) and a testing set (KDDTest+). Each sample in NSL-KDD composes of 41 features with an additional label. The samples can be labelled either as normal or one of the four main attack groups including *Denial of service (DoS)*, *Remote to Local (R2L)*, *User to Local (U2R)*, and *Probe*. Among the four attack groups, R2L was considered to be one of the most challenging for machine learning algorithms to detect [24]. R2L is embedded in their data packets itself, and does not form sequential patterns like DoS and Probe. Therefore, the network traffic of R2L tends to be very similar to the normal network traffic. DSAE is designed to perform efficiently on this attack group. For R2L, there are 995 instances in *KDDTrain+* and 2887 instances in *KDDTest+*.

- UNSW-NB15 [25] published in 2015 consists of the training set and the testing set. Each sample in UNSW-NB15 composes of 47 features and a label. There are nine attack types in UNSW-NB15 including *fuzzers*, *analysis*, *backdoor*, *DoS*, *exploit*, *generic*, *reconnaissance*, *shellcode*, and *worm*.

- The CTU13 [16] is a botnet dataset that consists of thirteen botnet scenarios. In this paper, we only use four scenarios in CTU13, and split each of them into 40% for training (normal traffic) and 60% for evaluating (normal and botnet traffic) [22].

All categorical features of three datasets are converted into real-values using one-hot-encoding. This results in higher dimensional versions of the tested data (the second column in Table 1).

5.2. Parameter settings

The settings for the hyper-parameters in DSAE are similar to the previous works [14, 9]. The number of hidden layers is five [14], the size of the bottleneck layer, m is chosen using the rule [9], $m = [1 + \sqrt{n}]$, where n is the number of input features, the batch size is 100, the hyperbolic tangent function is applied to all layers in DSAE except the bottleneck in which the sigmoi function is used. The loss function of DSAE in Equation (3) is optimized by using an adaptive SGD algorithm (ADADELTA) over 1000 epochs.

We conducted two experiments to examine the characteristics of DSAE. The first experiment is to compare the performance of DSAE to the previous model including SAE [11] and Denoising AutoEncoder (DAE) [36]. The second one is to assess the ability of DSAE in detecting the attacks that are difficult for SAE.

To measure the performance of the tested methods, we used the Area Under the ROC Curve (AUC) metric. The higher value of AUC a model produces, the better performance the model has. For each dataset, we executed the algorithms 10 times and the average value of AUC is reported.

6. RESULTS AND DISCUSSION

The first experiment aims to compare the performance of DSAE with SAE and Denoising AutoEncoder (DAE) on six well-known anomaly detection datasets. The results are presented in Table 2. In this table, we present the result of two version of DAE and SAE. One version (DAE+RE and SAE+RE) uses RE as the score for anomaly detection. The other version applies CEN on the latent representation of DAE and SAE. For DSAE we only present the results of DSAE+CEN since the version of DSAE+RE is less convincing. It can be seen that the AUCs of SAE and DSAE are mostly equal on the tested datasets and these values are often better than that of DAE. This result confirms that our proposed model performs comparatively in comparison to SAE and it is better than DAE. The second experiment is to compare the performance of DSAE and SAE on the dataset that

Table 2. AUCs from DAE, SAE, DSAE on six datasets

Tested Anomaly Model	**Datasets**					
	NSLKDD	**UNSW**	**CTU13-08**	**CTU13-09**	**CTU13-10**	**CTU13-13**
DAE* + CEN	0.854 ±0.002	0.690 ±0.001	0.938 ±0.015	0.655±0.031	0.951±0.006	0.711±0.002
DAE+RE	0.930±0.090	0.873±0.004	0.960±0.011	0.903±0.002	0.958±0.004	0.952±0.010
SAE**+CEN	0.960 ±0.002	0.896 ±0.006	0.982 ±0.009	0.940 ±0.010	0.997 ±0.001	0.964 ±0.012
SAE+RE	0.920 ±0.000	0.810 ±0.001	0.951 ±0.013	0.703 ±0.020	0.997 ±0.000	0.887 ±0.005
DSAE + CEN	**0.963** ±0.004	0.895 ±0.015	**0.986** ±0.012	0.929 ±0.054	0.992 ±0.008	**0.971** ±0.006

* DAE: Denoising AutoEncoder; ** SAE: Shrink AutoEncoder [11]

is difficult for SAE. Table 3 presents the results of DSAE and SAE on four attack groups of NSL-KDD. This table shows that DSAE is competitive with SAE on three attack groups (Probe, DoS, and U2R). However, on the most difficult attacks, i.e., the R2L attack group, DSAE (AUC of 0.936) out-performs SAE (AUC of 0.924). This result demonstrates that DSAE improve the performance of SAE in detecting the difficult attacks.

Table 3. AUCs from SAE, DSAE on four attack groups of NSL-KDD dataset

Tested Anomaly Model	**Datasets**			
	Probe	**DoS**	**R2L**	**U2R**
SAE + CEN *	0.977 ±0.003	0.967 ±0.002	0.924 ±0.010	0.956 ±0.005
DSAE + CEN	0.979 ±0.006	0.966 ±0.007	**0.936** ±0.011	0.960 ±0.010

* SAE + CEN: Shrink AutoEncoder and Centroid [11]

Table 4. Comparison the Detection between SAE and DSAE model on R2L attack group

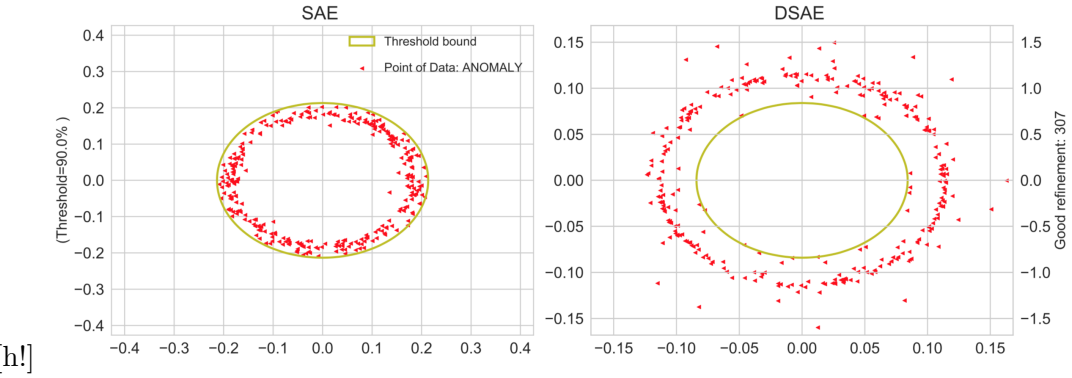
Tested Anomaly Model	**R2L Attack group**					
	TP	**FP**	**FN**	**TN**	**FAR**	**DETECTION RATE**
SAE + CEN *	1892	995	1008	8702	0.104	0.655
DSAE + CEN	2011	876	989	8721	**0.102**	**0.697**

* SAE + CEN: Shrink AutoEncoder and Centroid [11]

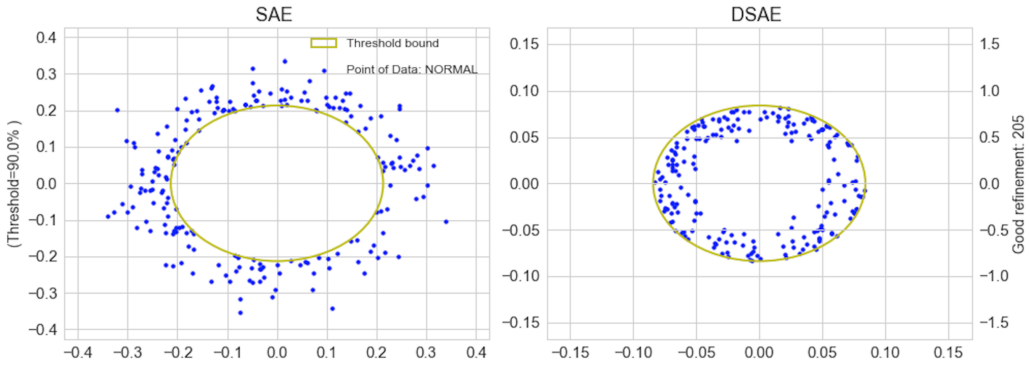
Table 4 shows the false alarm rate (FAR) and the detection rate (DR) of SAE and DSAE on R2L attack group. This result is calculated using the threshold equal 0.9. In other words, the threshold is selected so that 90% of samples are predicted as normal and the rest is predicted as abnormal. It can be seen from this table that on R2L attack group, DSAE is better than SAE on both metrics. More precisely, the FAR of DSAE is slighly lower than that of SAE (0.102 vs 0.104) and DR of DSAE is much higher than SAE (0.697 vs 0.655). Overall, the results in this section show that our proposed model, DSAE, performs equally effectively on the easy IDS datasets and it is much better than the state-of-the-art model, i.e. SAE, on the most difficult attack group.

To show how DSAE performs better than SAE on R2L, we carried out some investigations on latent vectors (both normal and anomalies) that are assigned different labels by SAE and DSAE. Since, the dimension of the latent vectors of SAE and DSAE is higher than two, we convert these vectors into two dimensions by calculating their Euclidean distances to the origin in the latent representation. The coordinates, x_i and y_i of each latent vector z_i are then computed as in Algorithm 1. We first visualize normal and anomaly data samples that are miss-classified by SAE, but correctly classified by DSAE in Figs. 2a and 2b. The data samples inside the yellow circle (a threshold) are classified as normal data, and outside are considered as anomalies. The threshold are selected so that DSAE can correctly classify 90%, 95% and 97% the normal training data. In the visualization, we set the threshold at 90%. Second, we plot the data samples that are correctly identified by SAE, but miss-classified by DSAE shown in Figs. 2a and Figs. 2c and 2d. It can be seen from these figures that DSAE performs very efficiently. Many data samples miss-classified by SAE are correctly predicted by DSAE. However, the second shrink pressure in DSAE also leads to some data samples (both normal and abnormal) that have been correctly identified by the first shrink being wrongly classified. Nevertheless, the figure shows that the number of incorrect samples by DSAE is much less than the number of incorrect samples by SAE particularly with normal samples (Fig. 2d).

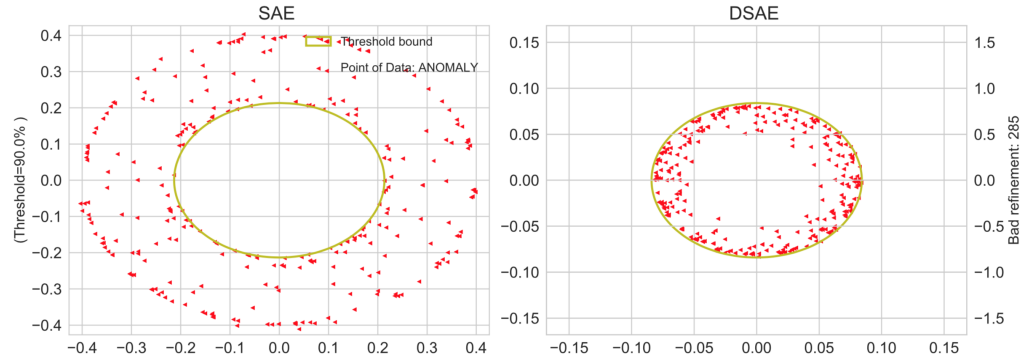
[h!]



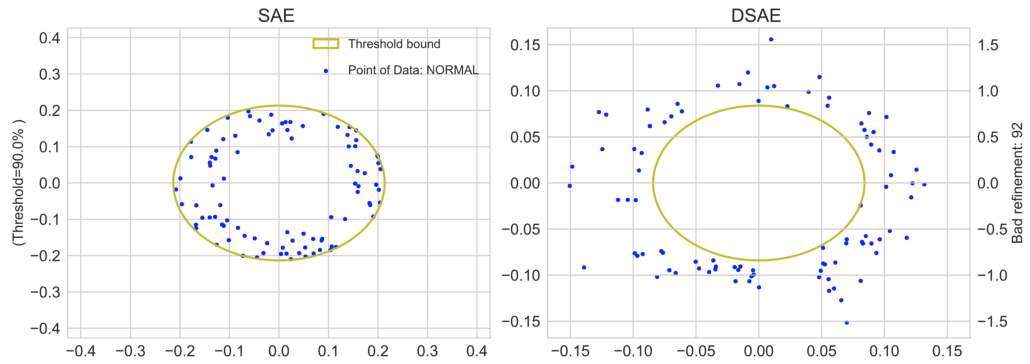
(a) Anomaly points are identified incorrectly by SAE, but correctly classified by DSAE



(b) Normal points are identified incorrectly by SAE, but correctly classified by DSAE



(c) Anomaly points are already identified correctly by SAE, but incorrectly classified by DSAE



(d) Normal points are already identified correctly by SAE, but incorrectly classified by DSAE

Figure 2. Monitoring points are identified incorrectly or correctly by SAE (left), but correctly or incorrectly classified by DSAE (right) using Visualize latent vectors in 2D

Algorithm 1 Visualize latent vectors in 2D

Input: a set of latent vectors (Z)
Output: a set of point $(x_i, y_i) \in P$ in 2D

Calculate Euclidean distance (dz_i) for each point $z_i \in Z$ to the origin.
$P \leftarrow \{\}$
for $i = 0$ to the size of Z **do**
    get α is random number between 0 to 360
    $x_i \leftarrow \cos\alpha * dz_i$
    $y_i \leftarrow \sin\alpha * dz_i$
    add $p(x_i, y_i)$ to P
end for
return P

7. CONCLUSIONS AND FUTURE WORK

This paper proposed a novel AE-based model, namely double-shrink autoencoder (DSAE) for network anomaly detection. DSAE aims to increase the ability in detecting the difficult detection attacks, such as the R2L attack group in NSL-KDD, whose network traffic is very similar to the normal traffic. The main idea in DSAE is to use double shrink to separate these types of attacks from the normal data. We conducted extensive experiments on six benchmark datasets in the domain of network anomaly detection. These consist of comparing the performance of DSAE against SAE and DAE, evaluating the ability of DSAE in detecting specific attack group (e.i R2L). The experimental results confirm that our proposed model not only performs comparatively to SAE, but also tends to out-perform SAE on the network traffic associated with the attacks tending to be very similar to the traffic of normal data, such as R2L.

There are a number of research directions for future work. Firstly, the normal network data may be presented in several group of similar of data (or clusters), thus finding clusters in the normal data may be a good processed step for the later learning processes of SAE. This means that we can develop SAE to learn multi-clusters in the latent feature space instead of only one cluster. Second, we will evaluate DSAE on a wide range of anomaly detection problems as well as synthetic data to demonstrate the ability in detecting the attack group whose behavior is very similar to normal behavior.

ACKNOWLEDGEMENT

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.05-2019.05.

REFERENCES

[1] I. Ahmad, A. B. Abdullah, and A. S. Alghamdi, "Remote to local attack detection using supervised neural network," in *2010 International Conference for Internet Technology and Secured Transactions*. IEEE, 2010, pp. 1–6.

[2] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016.

[3] D. K. Bhattacharyya and J. K. Kalita, *Network anomaly detection: A machine learning perspective*. Chapman and Hall/CRC, 2013.

[4] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition," *Biological Cybernetics*, vol. 59, no. 4, pp. 291–294, 1988.

[5] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *ACM Sigmod Record*, vol. 29, no. 2. ACM, 2000, pp. 93–104.

[6] T. C. Bui, L. V. Cao, M. Hoang, and Q. U. Nguyen, "A clustering-based shrink autoencoder for detecting anomalies in intrusion detection systems," in *2019 11th International Conference on Knowledge and Systems Engineering (KSE)*. IEEE, 2019, pp. 1–5.

[7] T. C. Bui, M. Hoang, and Q. U. Nguyen, "Some common datasets of a intrusion detection system and clustering properties," *Vietnam Journal of Science and Technology*, vol. 62, no. 1, pp. 1–7, 2020.

[8] T. C. Bui, M. Hoang, Q. U. Nguyen, and C. L. Van, "Data fusion-based network anomaly detection towards evidence theory," in *The 2019 6th NAFOSTED Conference on Information and Computer Science (NICS19)*. IEEE, 2019, pp. 33–38.

[9] V. L. Cao, M. Nicolau, and J. McDermott, "A hybrid autoencoder and density estimation model for anomaly detection," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2016, pp. 717–726.

[10] ——, "One-class classification for anomaly detection with kernel density estimation and genetic programming," in *European Conference on Genetic Programming*. Springer, 2016, pp. 3–18.

[11] ——, "Learning neural representations for network anomaly detection." *IEEE Transactions on Cybernetics*, vol. 49, no. 8, pp. 3074–3087, 2019.

[12] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[13] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 15:1–15:58, Jul. 2009. [Online]. Available: http://doi.acm.org/10.1145/1541880.1541882

[14] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, "High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning," *Pattern Recognition*, vol. 58, pp. 121–134, 2016.

[15] U. Fiore, F. Palmieri, A. Castiglione, and A. De Santis, "Network anomaly detection with the restricted boltzmann machine," *Neurocomputing*, vol. 122, pp. 13–23, 2013.

[16] S. Garcia, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *Computers & Security*, vol. 45, pp. 100–123, 2014.

[17] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers & Security*, vol. 28, no. 1-2, pp. 18–28, 2009.

[18] S. Hawkins, H. He, G. Williams, and R. Baxter, "Outlier detection using replicator neural networks," in *International Conference on Data Warehousing and Knowledge Discovery*. Springer, 2002, pp. 170–180.

[19] G. E. Hinton and R. S. Zemel, "Autoencoders, minimum description length and helmholtz free energy," in *Advances in Neural Information Processing Systems*, 1994, pp. 3–10.

[20] N. Japkowicz, C. Myers, M. Gluck *et al.*, "A novelty detection approach to classification," in *IJCAI*, vol. 1, 1995, pp. 518–523.

[21] P. G. Jeya, M. Ravichandran, and C. Ravichandran, "Efficient classifier for R2L and U2R attacks," *International Journal of Computer Applications*, vol. 45, no. 21, pp. 28–32, 2012.

[22] D. C. Le, A. N. Zincir-Heywood, and M. I. Heywood, "Data analytics on network traffic flows for botnet behaviour detection," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2016, pp. 1–7.

[23] W. Lee, S. J. Stolfo, and K. W. Mok, "A data mining framework for building intrusion detection models," in *Proceedings of the 1999 IEEE Symposium on Security and Privacy (Cat. No. 99CB36344)*. IEEE, 1999, pp. 120–132.

[24] N. Maček and M. Milosavljević, "Reducing U2R and R2l category false negative rates with support vector machines," *Serbian Journal of Electrical Engineering*, vol. 11, no. 1, pp. 175–188, 2014.

[25] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*. IEEE, 2015, pp. 1–6.

[26] M. M. Moya, M. W. Koch, and L. D. Hostetler, "One-class classifier networks for target recognition applications," *NASA STI/Recon Technical Report N*, vol. 93, 1993.

[27] D. Rajashekar, A. N. Zincir-Heywood, and M. I. Heywood, "Smart phone user behaviour characterization based on autoencoders and self organizing maps," in *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2016, pp. 319–326.

[28] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *International Conference on Machine Learning*, 2018, pp. 4393–4402.

[29] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*. ACM, 2014, p. 4.

[30] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[31] A. G. Tartakovsky, A. S. Polunchenko, and G. Sokolov, "Efficient computer network anomaly detection by changepoint detection methods," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 1, pp. 4–11, 2012.

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

[33] N. N. Thi, V. L. Cao, and N.-A. Le-Khac, "One-class collective anomaly detection based on lstm-rnns," in *Transactions on Large-Scale Data-and Knowledge-Centered Systems XXXVI*. Springer, 2017, pp. 73–85.

[34] T. van Laarhoven, "L2 regularization versus batch and weight normalization," *arXiv preprint arXiv:1706.05350*, 2017.

[35] K. Veeramachaneni, I. Arnaldo, V. Korrapati, C. Bassias, and K. Li, "AIˆ 2: training a big data machine to defend," in *2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS)*. IEEE, 2016, pp. 49–54.

[36] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th International Conference on Machine Learning*. ACM, 2008, pp. 1096–1103.

Received on November 04, 2019
Revised on March 13, 2020