Clustering-Based Deep Autoencoders for Network Anomaly Detection

Chapter · November 2020

DOI: 10.1007/978-3-030-63924-2_17

4 authors, including:

Nguyen Viet Hung
Le Quy Don Technical University
10 PUBLICATIONS **27** CITATIONS
SEE PROFILE

Nhien-An Le-Khac
University College Dublin
262 PUBLICATIONS **2,180** CITATIONS
SEE PROFILE

Van Loi Cao
University College Dublin
25 PUBLICATIONS **469** CITATIONS
SEE PROFILE

Clustering-based Deep Autoencoders for Network Anomaly Detection

Van Quan Nguyen[1], Viet Hung Nguyen[1], Nhien-An Le-Khac[2], and Van Loi Cao[1]

[1] Le Quy Don Technical University, Vietnam
[2] University College Dublin, Dublin, Ireland
nguyenvanquan87@mail.ru,hungnv@lqdtu.edu.vn,loi.cao@lqdtu.edu.vn
an.lekhac@ucd.ie

Abstract. A novel hybrid approach between clustering methods and autoencoders (AEs) is introduced for detecting network anomalies in a semi-supervised manner. A previous work has developed regularized AEs, namely Shrink AE (SAE) and Dirac Delta Variational AE (DVAE) that learn to represent normal data into a very small region being close to the origin in their middle hidden layers (latent representation). This work based on the assumption that normal data points may share some common characteristics, so they can be forced to distribute in a small single cluster. In some scenarios, however, normal network data may contain data from very different network services, which may result in a number of clusters in the normal data. Our proposed hybrid model attempts to automatically discover these clusters in the normal data in the latent representation of AEs. At each iteration, an AE learns to map normal data into the latent representation while a clustering method tries to discover clusters in the latent normal data and force them being close together. The co-training strategy can help to reveal true clusters in normal data. When a querying data point coming, it is first mapped into the latent representation of the AE, and its distance to the closest cluster center can be used as an anomaly score. The higher anomaly score a data point has, the more likely it is anomaly. The method is evaluated with four scenarios in the CTU13 dataset, and experiments illustrate that the proposed hybrid model often out-performs SAE on three out of four scenarios.

Keywords: Autoencoder, Deep learning, Anomaly Detection, Clustering, Latent Representation

1 Introduction

Anomaly detection is the task to identify patterns in data or events representing the operation of systems that vary so much from the expected behavior [1, 6]. In network security, the network anomaly detection means the discrimination of illegal, malicious activities and other damaging forms of network use and abuse from normal connections or expected behavior of network systems [13]. These

actions are considered network anomalies. In many scenarios, data representing the normal behavior of network systems tend to be available and easy to obtain, while anomalies are scarce and sometimes impossible to collect [5]. Thus, the semi-supervised learning approach, such as one-class classification techniques, is a suitable learning scheme to construct models from only normal data for identifying network anomalies. In such approach, One-class Support Vector Machine (OCSVM), Local Outlier Factor (LoF), and Autoencoders (AEs) are well-known methods used for anomaly detection.

There has been been a widespread use of AEs in anomaly detection domain [8, 12, 10, 14] in recent years. This AE method can be used to build anomaly detectors by itself, or used as representation blocks in hybrid anomaly detection models. In the latter approach, the bottleneck layer of the AE trained on normal data is used as the new feature representation (latent representation) for enhancing the performance of following anomaly detection methods. This is because the latent feature space can represent normal data in more meaningful features (lower dimension and reveal robust features) than the original feature space. The work in [5] is known as a typical study of using the latent representation. In [5], two regularized AEs, namely SAE and DVAE, were developed to construct a robust feature representation in which normal data is mapped into a very small region being close to the origin in the latent representation of SAE. The rest of latent feature space is reserved for possible anomalies appearing in the future. This work is based on the assumption that normal data points may share some common characteristics, and they can be forced to distribute in a small single cluster. In some scenarios, however, normal network data may contain data from very different network services, which may result in a number of clusters in the normal data. Therefore, representing such data into single cluster in the bottleneck layer of AE may damage valuable characteristics of normal data.

In this paper, we introduce a hybrid learning model that can inherited the strengthens from both autoencoders (AEs) and clustering methods for anomaly detection. In other words, AEs have the ability to map normal data in a lower feature space in which more robust features representing normal behavior can be revealed, whereas clustering methods can learn to discover sub-classes (i.e. clusters) in the normal data. When clustering methods working in a lower dimension with more robust features such the latent representation of AEs, they can perform more efficiently than those in the original input data. Therefore, this work aims to propose a novel learning scheme that combines the learning strategies of an AE and a clustering method at each iteration. This means that the AE learn to represent normal data into robust feature space in its middle hidden layer, while the clustering method automatically discovers a number of sub-classes and forces them being close together. This work is performed at each iteration until these training processes are stable. Our proposed model is novel to the exist hybrid between AEs and clustering techniques introduced in [15]. In [15], the authors aimed to discover a number of image clusters in the latent feature space of an AE in an unsupervised manner. The main difference is that

the hybrid in [15] was trained to pull each image cluster far away from each others while CAE learns to force normal clusters being close together because these normal clusters share some common characteristics of the normal behaviors. The proposed model is evaluated on the four scenarios in the CTU13 dataset. our experimental results show that CAE often out-performs SAE and DVAE on three of four scenarios where the normal data seems to have more than one classes.

The rest of this paper is organized as follows. We shortly introduce to AEs and a clustering method in Section 2. Section 3 briefly reviews some recent studies related to using latent representation for anomaly detection. This is followed by a section proposing the hybrid between AEs and clustering methods. Experiments, results and discussion are presented in Sections 5 and 6 respectively. The paper concludes with highlights and future directions.

2 Background

2.1 Autoencoders

A classical autoencoder [11, 2], often called autoencoder, is a neural network that consists of two parts: *encoder* and *decoder* as shown in Figure 1. An autoencoder is trained to copy network's input to its output. The *encoder* is defined as a feature extractor that allows to explicitly represent an input x in a feature space. Let f_θ denote the encoder, and $X = \{x_1, x_2, ...x_n\}$ be a dataset. The *encoder* f_θ map the input $x_i \in X$ into a latent representation $z_i = f_\theta(x_i)$. The *decoder* g_ϕ attempts to map the *latent representation* z_i back into the input space, which forms a reconstruction $\hat{x}_i = g_\phi(z_i)$. The encoder and decoder are commonly represented as single-layer neural networks in the form of activation functions of affine mappings as follows:

$$f_\theta(x) = \psi_f(Wx + b) \tag{1}$$

$$g_\phi(z) = \psi_g\left(W^{'}z + b^{'}\right) \tag{2}$$

where θ=(W,b) and ϕ =(W',b') are parameters set for training encoder and decoder, respectively. ψ_f and ψ_g are the activation functions of the encoder and decoder, such as a *logistic sigmoid* or *hyperbolic tangent* non-linear function, or a linear *identity* function. The reconstruction loss function over training samples can be written as:

$$\mathcal{L}_{\mathrm{AE}}(\theta; \phi; x) = \frac{1}{n}\sum_{i=0}^{n} l(x_i, \hat{x}_i) = \frac{1}{n}\sum_{i=0}^{n} l(x_i, g_\phi(f_\theta(x_i))) \tag{3}$$

Where $l(x_i, \hat{x}_i)$ is the discrepancy between the input x_i and its reconstruction $\hat{x}_i$; n is the number of data samples in the dataset. Autoencoders learn to optimize the objective function in (3) with respect to the parameters θ=(W,b) and ϕ =(W',b') by using a learning algorithm such as the stochastic gradient descent with back-propagation. The choice of the reconstruction loss depends largely

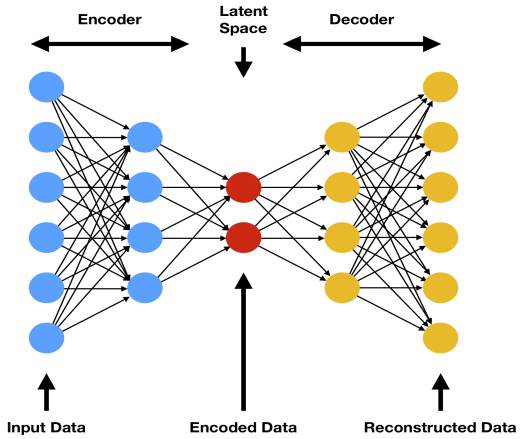


Fig. 1. Autoencoder

on the appropriate distributional assumptions on given data. The *mean squared error* (MSE) in Eq. 4 is commonly used for real-valued data, whereas a *cross-entropy* loss in Eq. 5 can be used for binary data.

$$\mathcal{L}_{\mathrm{AE}}(\theta; x) = \frac{1}{n} \sum_{i=1}^{n} \left(\parallel x_i - \hat{x}_i \parallel^2 \right) \tag{4}$$

$$\mathcal{L}_{\mathrm{AE}}(\theta; x) = -\frac{1}{n} \sum_{i=1}^{n} \left(x_i \log(\hat{x}_i) + (1 - x_i) \log(1 - \hat{x}_i) \right) \tag{5}$$

By compressing input data into a lower dimensional space, the classical autoencoder avoids simply learning the identity, and removes redundant information [12].

2.2 Clustering techniques

Clustering techniques consider samples in datasets as objects. They classify these objects into different groups or clusters, so that objects within a cluster are similar to each others and distinguished with objects from other clusters. There are variety type of clustering algorithms and every methodology follows a different set of rules for defining the similarity among data points. One of the most popular clustering algorithms is K-means. It was originally proposed by MacQueen in 1967. K-mean is a unsupervised, interative method of clustering. The idea behind this technique is that the algorithm starts with a given dataset and initial clustering centers. Following this, two steps are operated iteratively: relocating each data point to its new closest center; updating the clustering centers by calculating the mean of all members. The process is repeated until a convergence

criteria is met (a predefined number of iterations). If no changes in clustering centers between two consecutive iterations is found, the process will stop, and return the optimal solution. Pseudo-code of K-Means algorithm is showed in Algorithm 1.

Algorithm 1 K-Means

1: **Input: given a datasets** $X = \{x_1, x_2, ...x_n\}$
and the number of cluster k**.**
2: Output: k **cluster centers.**
3: Initialize: Randomly choose k **initial centers** $\{c_1, c_2, ...c_k\}$**.**
4: repeat
5: Assign each sample $x_i \in X$ **to the closest cluster.**
6: Calculate the new mean and update the new cluster center.
7: until a convergence criteria is met
=0

3 Related Work

Autoencoders (AEs) have been widely used for anomaly detection. The network architectures can be used by itself as a stand-alone classifier or a feature representation block in a hybrid between AEs and classification methods. In the latter approach, the bottleneck layer of AEs can be used as a new feature space for the following classifiers [5, 4, 3, 10, 7, 15, 16]. The latent feature representation can be constructed in supervised learning [16], semi-supervised learning [5, 4, 3] and unsupervised learning [10, 7]. Once the training process of a AE finished, the encoder is used as a feature representation block enhancing the following anomaly detection method. The central idea is that the bottleneck layer can map original data into a lower dimension, and discover more robust features representing the normal behavior. The remain of this section, we will discuss some typical work for three latent representation approaches mentioned above.

In supervised manner, Vu et al. [16] introduced Multi-distribution VAE (MVAE) to represent normal data and anomalous data into two different areas in the middle hidden layer of VAE. Classical Variational autoencoders (VAEs) can learn to map input data into a standard Gaussian distribution $\mathcal{N}(0, 1)$ in its bottleneck layer. In [16], the class labels (normal and anomaly) are incorporated into the loss function of VAE to push the two data classes into two different regions. These regions have the same Gaussian distribution shape with $\sigma = 1$, but different mean values. The proposed model was evaluated on two publicly network security datasets, and it produces promising performance.

In semi-supervised manner, Cao et al. [5] proposed two regularized AEs, called Shrink AE (SAE) and Dirac Delta VAE (DVAE), to capture the behaviors of normal data. SAE and DVAE are aimed to put normal data towards a very small region being close to the origin of the latent feature space, and attempt to reserve the rest of the latent feature space for anomalies appearing in the future.

The authors assumed that only normal samples are available for training, and no anomalous data can be used for estimating hyper-parameters. These regularized AEs are aimed to address the problem of identifying anomalies in high-dimensional network data. The latent representation of SAE and DVAE were then used for facilitating simple one-class classifiers. SAE and DVAE were then evaluated on eight well-known anomaly detection datasets. The experimental results confirmed that their models not only can produce a better performance, but also are less sensitive on a wide range of parameter settings in comparison to stand-alone OCCs, and those of other feature representation methods.

The study [7] is a typical unsupervised learning approach that employs the latent representation of an AE, namely a deep belief network (DBN). Erfani et al. [7] used a deep belief network (DBN) for constructing a robust feature representation for an one-class classifications (OCCs). The OCCs are One-class Support Vector Machine (OCSVM) and Support Vector Data Description (SVDD). The proposed model aimed to deal with the problem posed in high-dimensional anomaly detection data. The DBN was first pretrained in the greedy layer-wise fashion by stacking Restricted Boltzmann Machines (RBMs) trained in unsupervised manner. OCSVM and SVDD were then stacked on top of the DBN. The hybrid model were evaluated on eight high-dimensional UCI datasets, and the experimental results showed that the model often out-performs stand-alone the one-class classifiers.

In this work, we attempt to construct a latent representation for identifying network anomalies in semi-supervised manner. This means that we first train an AE on normal data to map the original data into several clusters in the latent feature space of the AE with a proposed regularized loss. It then performs the K-mean task: assigning data points to each clusters; relocating the cluster centers. These tasks are operated iteratively until a early-stopping criteria met. The trained CAE is then employed for classifying querying data points. The distance from the data point to the nearest cluster center is used as anomaly score. By imposing a classification threshold, the data point can be classified as an anomaly if its distance is greater than the threshold.

4 Proposed Approach

In this section, we explain our proposed model for anomaly detection that is called Clustering-based deep AutoEncoder (CAE). Our model is a combination of a variation of k-mean clustering algorithm and an AE. It is clear that the original version of AE, which forces normal input samples into a sole cluster may not produce expected results on datasets in which practically have several clusters. CAE aims to learn the latent representation of normal data and parallelly force data points in the bottleneck of AE into a number of clusters. During the training process, the centers of these clusters are also put toward to the origin of the latent representation. This is based on the assumption that normal data may have several sub-classes inside, but they still share some common characteristics in overall. Thus, normal data points should be appeared close each others.

In practice, we propose a new regularizer to the loss function of the AE in the hybrid CAE. Unlike classical K-mean methods, which try to minimize distances within a cluster and maximize the distances between clusters, we aim to design a variance K-mean that minimizes both distances within each cluster and between clusters. Therefore our regularizer consists of two terms: (1) minimizing the distances of data points to their corresponding centers; (2) minimizing the distances of the cluster centers to the origin of the latent representation. In the equation 3, the first term is the reconstruction error (RE) of the AE, and the second and the third terms are the two regularized terms (1) and (2) respectively. The CAE model trained on normal training data forces almost normal data into inner clusters and these cluster centers are also pulled closer to the origin of the CAE latent representation. In the querying stage, the model will be employed to distinguish whether a query data point belonging to normal class or anomalous class by evaluating its distance to the closest cluster center. This means that if the distance is greater than a predetermined threshold, the data point will be classified as an anomaly. The loss function in this case can be defined as in Equation 6 below:

$$\zeta_{CAE}(\theta, x^i, z) = \frac{1}{n}\sum_{1}^{n}\left|\left|x^i - \hat{x}^i\right|\right|^2 + \lambda_1 \frac{1}{n}\sum_{1}^{n}\left|\left|f^{(t)}(x_i) - c_i^*\right|\right|^2 + \lambda_2 \frac{1}{k}\sum_{1}^{k}\left|\left|c_j^t\right|\right|^2 \tag{6}$$

$$c_i^* = \underset{c_j^{t-1}}{argmin}\left|\left|f^{(t)}(x_i) - c_j^{t-1}\right|\right|^2 \tag{7}$$

$$c_j^t = \frac{\sum_{x_i \in C_j^{t-1}} f^t(x_i)}{\left|C_j^{t-1}\right|} \tag{8}$$

where n is the number of samples in the normal training set; $\hat{x}^i$ is the reconstruction of the sample x^i; The first component in (6) is the reconstruction error (RE), and the second and the third terms are two regularized components (pulling data points closer to their cluster centers, pushing the centers of all clusters being close to the origin as well as possible). λ_1 and λ_2 are used to trade-off between these components. $f^t(x)$ is the mapping function (the encoder of CAE) at the t^{th} iteration; c_i^* is the closest cluster of the i^{th} observation; c_j^{t-1} is the j^{th} cluster centroid, which is produced at the $(t-1)^{th}$ iteration.

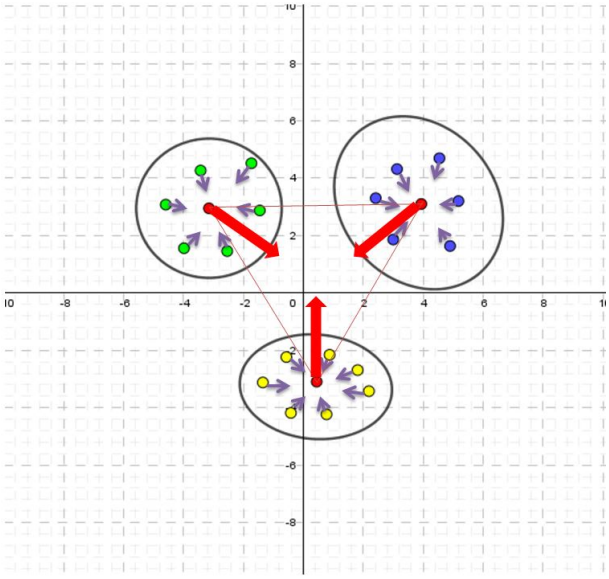


Fig. 2. An illustration of our proposed model

Fig. 2 is an example for representing noramal data in the latent feature space of CAE during the co-training process of CAE. We suppose that the normal training set consists of three clusters (illustrating in green, blue, and red). In the training phrase, observations in the same cluster are forced to be close to their corresponding cluster centers as showed in small purple arrows, meanwhile three cluster centers are also pushed towards the origin of the latent feature space as illustrated with red arrows.

We present our proposed method as in Algorithm 2. At the first iteration C^0, we randomly initialize K cluster centers. CAE is trained on the normal training set to optimize the loss function (6). This process will gradually force the normal training data in the latent feature space (called latent normal data) into several clusters. Following this, the latent normal data is assigned into K clusters via formula (7) and these cluster centers are updated using the equation (8). These training steps will be operated until an early-stopping criteria met or the iteration exceed a threshold T. In this work, we use a normal validation set for the early-stopping. The model trained on the normal training set is evaluated on the validation set once every few iterations. The training process will stop if the RE on the validation set does not improve for a certain threshold in a number of successive iterations. Finally, the output of the algorithm is the cluster centers in the latent feature space and CAE model.

Algorithm 2 Clustering-based deep AutoEncoder (CAE)

Input: a given dataset X, the number of clusters K, a set of AE hyper-parameters θ and ϕ, the maximum number of iterations T.
randomly choose K cluster centers (C^o)
$t = 1$
repeat
  Train CAE on X to optimize the loss function (6).
  Classify X into K clusters
  Update all new K cluster centers (C^t) **via formula (8).**
  $t = t + 1$**.**
until an early-stopping criteria met or $t < T$.
Output: CAE model and cluster centers C.
=0

5 Experiments

5.1 Datasets

In order to demonstrate the efficient performance of the proposed model, we conducted the experiments on four datasets as shown in the Table 1. These datasets are the mostly well-known issues in the cyber security domain, namely CTU13. The CTU13 is a publicly available botnet data provided in 2011 [9]. In each experiment we split them into 40% for training (normal traffic) and 60% for evaluating (normal and botnet traffic). Three categorical features, including *dTos, sTos* and *protocol* are encoded by the one-hot encoding technique. We follow the processing procedure in [5] to preprocessing these datasets.

Table 1. Four datasets for evaluating the proposed models

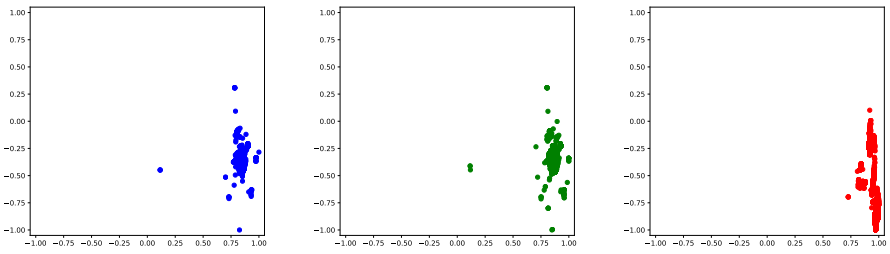
No	Dataset	Dimension	Training set	Normal Test	Anomaly Test
1	Rbot (CTU13-10)	38	6338	9509	63812
2	Murlo (CTU13-8)	40	29128	43694	3677
3	Neris (CTU13-9)	41	11986	17981	110993
4	Virut (CTU13-13)	40	12775	19164	24002

5.2 Experimental Settings

In this work, our model will be constructed from the normal class only. The configuration of CAE is described as follows. The trade-off parameters of the loss function λ_1 and λ_2 are set equal to 300 and 1500 respectively. The number of hidden layer is 5, and the middle hidden layer size is calculated using the formula $h = [1 + \sqrt{n}]$, where n is the number of input features [4]. The learning

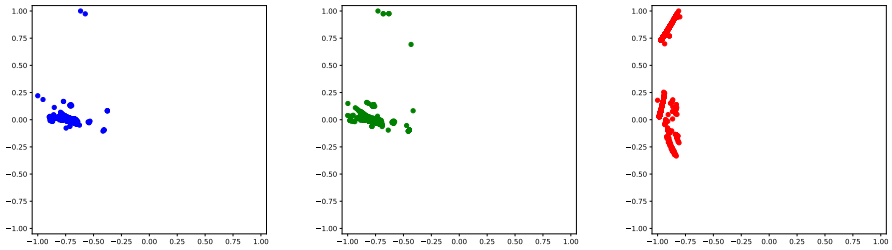
rate is set at 10^{-1}, and the batch size is 100. The weights of CAE are initialized using the Xavier initialization method to speed up the convergence process. We use *Adadelta* optimization algorithm for training CAE. The activation function is the TANH function. For the variance K-mean, we manually choose the number clusters equal to 2. We realize that $k = 2$ is the best choice, but it is acceptable. How to automatically estimate the number of sub-classes (clusters) in normal network data is a question for our future work. We use validation sets to evaluate our proposed models at every 5 epochs for early-stopping. The normal training data, normal testing data and anomaly testing data in the latent representation are visualized for getting into the insight of the latent feature space.

We carry out two experiments for evaluating the proposed model. Firstly, the performance of CAE is compared with SAE and DVAE in the study [5]. Therefore, we reproduce the same experiments as in [5], and report the performance of SAE and DVAE on two cases: the following classifiers are CEN and MDIS as showed in Table 2. The reason for choosing CEN and MDIS is that CEN was reported as the best classifier when combining with SAE and DVAE in the study [5] while MDIS was considered as the worse case. In our model, we use the distance from a querying data point to its nearest cluster center as anomaly a score. To estimate the distance, our model first calculate the distance from that point to every cluster centers, and decide which one is the nearest cluster center. The process is very similar to CEN proposed in [5], but for multiple centroids. We call the method to estimate anomaly score in our proposed model as Multi Centroids (M-CEN). Secondly, the latent data of the training set and testing set are visualized to get into insight the behavior of the latent representation of CAE as visualizing in Figs. 3, 6, 4, and 5. We also visualize the Are Under the ROC curves when evaluating CAE on four scenarios as showed in Fig. 7.



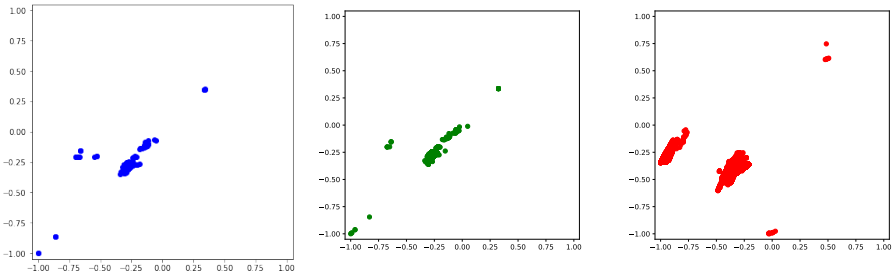
(a) Normal training (b) Normal testing (c) Anomaly testing

Fig. 3. Visualize the latent data of the CTU13-10 dataset



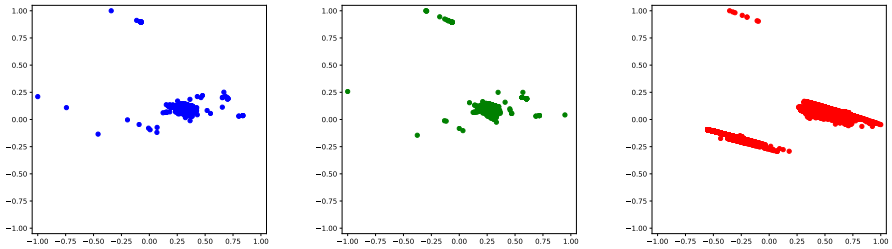
(a) Normal training (b) Normal testing (c) Anomaly testing

Fig. 4. Visualize the latent data of the CTU13-08 dataset



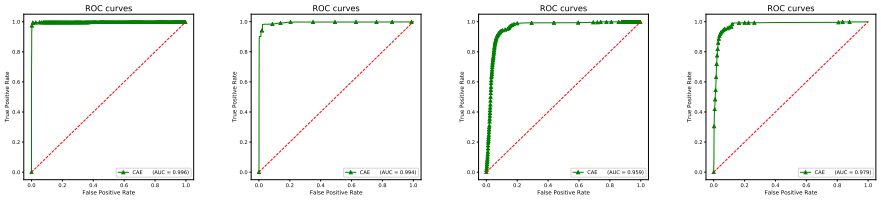
(a) Normal training (b) Normal testing (c) Anomaly testing

Fig. 5. Visualize the latent data of the CTU13-9 dataset



(a) Normal training (b) Normal testing (c) Anomaly testing

Fig. 6. Visualize the latent data of the CTU13-13 dataset



(a) CTU13-10 (b) CTU13-08 (c) CTU13-09 (d) CTU13-13

Fig. 7. The ROC curves of our proposed model on four datasets

Table 2. AUCs from the hybrid SAE-OCCs, DVAE-OCCs, and the CAE model.

Represen- **-tation**	**One-class Classifiers**	**Datasets**			
		CTU13-10	CTU13-08	CTU13-09	CTU13-13
SAE	CEN	0.999	0.991	0.950	0.969
$\lambda = 10$	MDIS	0.999	0.990	0.950	0.968
DVAE	CEN	0.999	0.982	0.956	0.963
$\lambda = 0.05$, $\alpha = 10^{-8}$	MDIS	0.999	0.984	0.957	0.964
CAE	M-CEN	0.996	**0.994**	**0.959**	**0.979**

6 Results and Discussion

This section presents the experimental results of evaluating the proposed model, CAE, on the four scenarios in the CTU13 dataset. The performance of these classifiers is evaluated by using AUC as summarized in Table 2. The training data and testing data represented in the latent feature space of CAE are plotted to investigate the behavior of CAE. The ROC curves produced from the CAE classifier are also visualized to evaluate the performance of CAE on a number of classification thresholds as showed in Fig. 7.

It can be seen from Table 2 that the CAE classifier performs (in terms of classification accuracy) very well on all datasets, and the CAE performance is often better than those of SAE and DVAE with CEN and MDIS on the CTU13-08, CTU13-09 and CTU13-13 scenarios. This suggests that the normal data may originally distribute in two sub-classes. Thus, when CAE represents the normal data into two clusters, the AUC scores are improved in comparison to those produced from SAE-CEN and DVAE-CEN.

In order to support for our discussion above, we will investigate the latent data of both the normal training data, and the testing data consisting of normal data and anomalies. In the figures 3, 6, 4, and 5, we plot the first two features of the data in the latent feature space: the normal training data in blue; the normal testing data in green; and the anomaly testing data in red. Figure 3 shows that both the normal training data and the normal testing data seem to distribute in only one region in the latent feature space. This suggests that the normal data in CTU13-10 may contain only one cluster. Therefore, the AUC

score on CTU13-10 decreases when the normal data is forced into two clusters (the AUC of CAE is slightly lower than those of SAE-CEN and DVAE-CEN). On the other hand, the figures 6, 4, and 5 show promising results. The latent data of the normal training, and the latent data of the normal testing distribute into two small areas in the latent feature space. These small areas tend to be close together. The latent data of the anomaly data appears in some regions not overlapping the normal data. This can imply that the normal data of CTU13-13, CTU13-08, and CTU13-09 may contain more than one sub-classes, which results in the increase in the AUC scores in comparison to those of SAE-CEN and DVAE-CEN as showed in the last row in Table 2.

The results in the figures 3, 6, 4, and 5 are very useful for the explanation why CAE can out-performs SAE and DVAE on the CTU13-13, CTU13-08, and CTU13-09 scenarios while on CTU13-10 it does not. Base on our experimental results and analysis, we can confirm that our proposed model, CAE, can perform well on anomaly detection datasets in which the normal data contains more than one sub-classes.

7 Conclusion and Future work

A novel hybrid between clustering methods and AEs, namely CAE, is proposed for automatically revealing a number of clusters in normal data. This aims to overcome the drawback of a previous study [5] where normal data is considered as only one cluster. In CAE, a variation of K-mean and an AE co-train at each training iteration in which the AE maps normal data into the latent representation while the clustering-based method attempts to group the data into several clusters being close to the origin. In querying stage, the distance of a given data point to the closest cluster center can be used as an anomaly score. The higher anomaly score a data point has, the more likely it is anomaly. CAE is evaluated on four scenarios in the CTU13 dataset and experiments illustrate that CAE out-performs SAE and DVAE on three out of four scenarios.

In this work, we choose the number of clusters manually. The study of automatically estimating the true number of clusters in normal data will be postponed to the future work.

References

1. Aggarwal, C.C.: Outlier analysis. In: Data Mining. pp. 237–263. Springer (2015)
2. Bourlard, H., Kamp, Y.: Auto-association by multilayer perceptrons and singular value decomposition. Biological Cybernetics **59**(4), 291–294 (1988)
3. Bui, T.C., Cao, V.L., Hoang, M., Nguyen, Q.U.: A clustering-based shrink autoencoder for detecting anomalies in intrusion detection systems. In: Proc. KSE. pp. 1–5. IEEE (2019)
4. Cao, V.L., Nicolau, M., McDermott, J.: A hybrid autoencoder and density estimation model for anomaly detection. In: International Conference on Parallel Problem Solving from Nature. pp. 717–726. Springer (2016)

5. Cao, V.L., Nicolau, M., McDermott, J.: Learning neural representations for network anomaly detection. IEEE transactions on cybernetics **49**(8), 3074–3087 (2018)
6. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM computing surveys (CSUR) **41**(3), 15 (2009)
7. Erfani, S.M., Rajasegarar, S., Karunasekera, S., Leckie, C.: High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning. Pattern Recognition **58**, 121–134 (2016)
8. Fiore, U., Palmieri, F., Castiglione, A., De Santis, A.: Network anomaly detection with the Restricted Boltzmann Machine. Neurocomputing **122**, 13–23 (2013)
9. Garcia, S., Grill, M., Stiborek, J., Zunino, A.: An empirical comparison of botnet detection methods. Computers & Security **45**, 100–123 (2014)
10. Hawkins, S., He, H., Williams, G., Baxter, R.: Outlier detection using replicator neural networks. In: International Conference on Data Warehousing and Knowledge Discovery. pp. 170–180. Springer (2002)
11. Hinton, G.E., Zemel, R.S.: Autoencoders, minimum description length and Helmholtz free energy. In: Advances in Neural Information Processing Systems. pp. 3–10 (1994)
12. Japkowicz, N., Myers, C., Gluck, M.: A novelty detection approach to classification. In: IJCAI. pp. 518–523 (1995)
13. Phoha, V.V.: Internet security dictionary. Springer Science & Business Media (2007)
14. Sakurada, M., Yairi, T.: Anomaly detection using autoencoders with nonlinear dimensionality reduction. In: Proc MLSDA. p. 4. ACM (2014)
15. Song, C., Liu, F., Huang, Y., Wang, L., Tan, T.: Auto-encoder based data clustering. In: Iberoamerican congress on pattern recognition. pp. 117–124. Springer (2013)
16. Vu, L., Cao, V.L., Nguyen, Q.U., Nguyen, D.N., Hoang, D.T., Dutkiewicz, E.: Learning latent distribution for distinguishing network traffic in intrusion detection system. In: ICC 2019-2019 IEEE International Conference on Communications (ICC). pp. 1–6. IEEE (2019)