A Hybrid Autoencoder and Density Estimation Model for Anomaly Detection

Van Loi Cao(✉), Miguel Nicolau, and James McDermott

NCRA Group, University College Dublin, Dublin, Ireland
loi.cao@ucdconnect.ie, {miguel.nicolau,james.mcdermott2}@ucd.ie
http://ncra.ucd.ie

Abstract. A novel one-class learning approach is proposed for network anomaly detection based on combining autoencoders and density estimation. An autoencoder attempts to reproduce the input data in the output layer. The smaller hidden layer becomes a bottleneck, forming a compressed representation of the data. It is now proposed to take low density in the hidden layer as indicating an anomaly. We study two possibilities for modelling density: a single Gaussian, and a full kernel density estimation. The methods are tested on the NSL-KDD dataset, and experiments show that the proposed methods out-perform best-known results on three out of four sub-datasets.

Keywords: Anomaly detection · Autoencoder · Density estimation

1 Introduction

Anomaly detection plays an important role in a variety of application domains ranging from intrusion detection in network security, credit card fraud detection, health care and insurance to fault detection in safety critical systems [1,3]. This is due to the fact that anomalies often translate to critical, actionable information or potentially dangerous situations and events. In network security, anomaly detection is the task of distinguishing illegal, malicious activities from normal traffic or behavior of systems [3,13]. This has become increasingly important due to valuable resources and the widespread use of computer networks in recent years.

Network anomaly detection models must be sufficiently flexible to keep up with the continuous evolution of attacks or malicious activities over time, and the occurrence of new, unknown anomalies [8]. Moreover, labeled anomaly data may not be available, due to the rarity of intrusions, difficulty of labeling, and the privacy and security concerns of computer networks [8,19]. For these reasons, one-class learning or novelty detection is a common approach for network anomaly detection. A one-class classifier constructed from only normal (target) data is employed to classify whether an unseen instance belongs to the normal class or anomaly (non-target) class [15].

J. Handl et al. (Eds.): PPSN XIV 2016, LNCS 9921, pp. 717–726, 2016.
DOI: 10.1007/978-3-319-45823-6_67

We are continuing previous research on one-class classification (OCC) with Kernel Density Estimation (KDE) [2]. It works by defining a threshold on *density* of the normal data: query points below the threshold are classed as anomalies. Several approaches to modeling density are possible, e.g. a single Gaussian, multiple independent Gaussians, and negative mean distance [21], but KDE is the most flexible of all. We found that KDE performed very well (better than One-class SVM [18]), but was slow at query time, so we provided a method to speed it up using Genetic Programming (GP).

Another method commonly used for anomaly detection is autoencoders (AEs). This design was named an "autoencoder" by Japkowicz et al. [11], who applied it for novelty detection in 1995. An autoencoder is a neural network which learns to reconstruct its input at the output layer. A narrow middle layer compresses redundancies in the input data while non-redundant information remains [11]. The effect is rather like a non-linear PCA. AEs are commonly used as building blocks in deep neural networks [10], and a key idea is that after training, the output layer is discarded, and the hidden layer is used as a new feature representation. In the one-class learning context, the reconstruction error (RE) of trained AEs is commonly used as a measure of "anomalyness".

In this paper, we investigate the distribution of data in the AE hidden layer. Based on this, we will propose a novel one-class learning method which models density of the compressed data from hidden layer on a trained AE. Two well-known density estimators are employed to model the density from hidden layer, a single Gaussian and a full KDE. An autoencoder is first trained on the normal class to minimize RE. The normal data is then passed through the trained AE again, and its density in the hidden layer is estimated. At the testing stage, a query point is first passed through the trained AE, and its value at the hidden layer is classified into normal or anomaly class by the density models.

The rest of this paper is organized as follows. We briefly review some work related to OCC based on AEs. In Sect. 3, we give a short introduction to AEs and density estimation. This is followed by a section proposing OCC using AEs and density estimation together. Experiments, Results and Discussion are presented in Sects. 5 and 6 respectively. The paper concludes with highlights and future directions.

2 Related Work

Recently, autoencoders or bottleneck neural networks became popular for anomaly detection as one-class learning techniques [17,22]. Hawkins et al. [9] trained a replicator neural network with narrow middle layers on normal data to construct a one-class classifier using reconstruction error as an indicator of anomalies. They used a step-wise activation function for the hidden layer to divide the continuously distributed data into clusters. Similarly, Sakurada and Yairi [17] compared classifiers based on AE, denoising AE, linear PCA, and kernel PCA. The classifiers were evaluated on spacecraft telemetry data. The learned features in the hidden layer were also examined.

Veeramachaneni et al. [22] proposed an ensemble learner to combine three single classifiers: AE, density-based, and matrix decomposition-based. They also used a human expert to provide ongoing correct labels for the algorithms to learn from. They tested their model on a large network log file dataset, with good results.

Erfani et al. [7] proposed a hybrid of a Deep Belief Network (DBN) and a linear one-class SVM for high-dimensional anomaly detection. A one-class SVM was built on the top of the trained DBN. This structure takes advantage of high decision classification accuracy from one-class SVMs and non-linear feature reduction from DBNs. The model was tested on eight UCI datasets, with comparable results to AE, and a significant improvement at query time.

In our work, we present a new approach for anomaly detection. We apply density estimation on the compressed data in the hidden layer. This method is distinct from those discussed above.

3 Preliminaries

3.1 Autoencoder

An autoencoder is a neural network with a (typically) narrow middle layer ("bottleneck"). It attempts to reproduce the input at the output, as illustrated in Fig. 1(a). It is commonly used for novelty detection and deep learning [9,11].

Let $x \in \mathbb{R}^n$ be an input example. The hidden representation $z(x) \in \mathbb{R}^m$ is represented in Eq. 1,

$$z\left(x\right) = f_1\left(W_1 x + b_1\right) \tag{1}$$

where f_1 is a non-linear activation function, $W_1 \in \mathbb{R}^{n \times m}$ is a weight matrix, $b_1 \in \mathbb{R}^m$ is a bias vector. The latent representation z is then mapped back into a reconstruction $\hat{x} \in \mathbb{R}^n$ in the output layer:

$$\hat{x} = f_2\left(W_2 z(x) + b_2\right) \tag{2}$$

where $W_2 \in m \times n$ and $b_2 \in \mathbb{R}^n$ are the weight matrix and bias vector of the output layer. f_2 is the output function. In this work, the logistic function (Eq. 3) and the identity function are used for hidden and output layers respectively. In Eq. 3, k is a steepness parameter.

$$f_1\left(z\right) = \frac{1}{1 + e^{(-kz)}} \tag{3}$$

The parameters of the network, $\theta = \{W_1, W_2, b_1, b_2\}$, are optimized such that the average reconstruction error (RE) is minimized. RE can be measured in many ways, and mean square error (MSE) is commonly used in training neural networks. In order to minimise the RE, stochastic gradient descent (SGD) is commonly used to train the network.

For anomaly detection, a model trained on normal data tends to fail to reproduce anomaly data, and produces high RE. Therefore, the reconstruction error is used as anomaly score. A test instance will be regarded as an anomaly if its RE is higher than a pre-determined error threshold.

3.2 Density Estimation

In this section, we briefly describe two methods of estimating density, *Centroid* and KDE. The Centroid method uses a single Gaussian, whose mean is placed at the centroid of the training data. The standard deviation is chosen to equal the standard deviation of the data, but in fact is unimportant: when we impose a threshold on density, the method becomes equivalent to imposing a threshold on distance (i.e. radius) from the centroid.

KDE is a non-parametric method of estimating probability density given a sample. Let $x_1, x_2, ..., x_n$ be a set of d-dimensional samples in $\mathbb{R}^d$ drawn from an unknown distribution with density function $p(x)$. An estimate $\hat{p}(x)$ of the density at x can be calculated using

$$\hat{p}(x) = \frac{1}{n} \sum_{i=1}^{n} K_h (x - x_i) \tag{4}$$

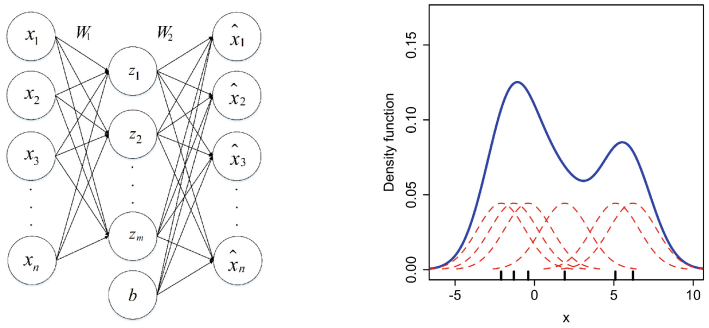
where $K_h : \mathbb{R}^d \rightarrow \mathbb{R}$ is a kernel function with a parameter h called the *bandwidth*. The Gaussian kernel (Eq. 5) is common in applications and is the only one used in this paper. As illustrated in Fig. 1(b) in KDE each point contributes a small "bump" to the overall density, with its shape controlled by the kernel and bandwidth. The bandwidth parameter h controls the trade-off between bias of the estimator and its variance.

$$K_h (x) = \exp (-\frac{x^2}{2h^2}) \tag{5}$$

4 Proposed Approach

Our proposed approach is to use density estimation on the hidden layer of an autoencoder. Our motivation for this is the same as that for RE-based OCC: anomaly data is poorly reconstructed by an AE trained on normal data, and part of this must be due to anomaly data occupying an unusual position in the hidden layer. We demonstrate this in Fig. 3. There are two phases in our method, training and testing, as illustrated in Fig. 2. In the training phase, an AE is first trained on a normal training set, and the training set is then passed through the trained AE again. The training data, compressed in the hidden layer, is used to build a density model. Based on the training stage, a density threshold is set, for example keeping 95 % of the training set. The compressed data will be classified as normal or anomaly by a threshold on the density model. Two density estimation methods are employed: Centroid and KDE.

The combination of an AE and density estimation takes advantage of their different strengths. AEs can compress input data to fewer dimensions while retaining non-redundant information, while density estimation works best in lower-dimensional spaces.



(a) (b)

Fig. 1. (a) An autoencoder. (b) Density estimated by KDE (Figure from https://en.wikipedia.org/w/index.php?title=Kernel_density_estimation)

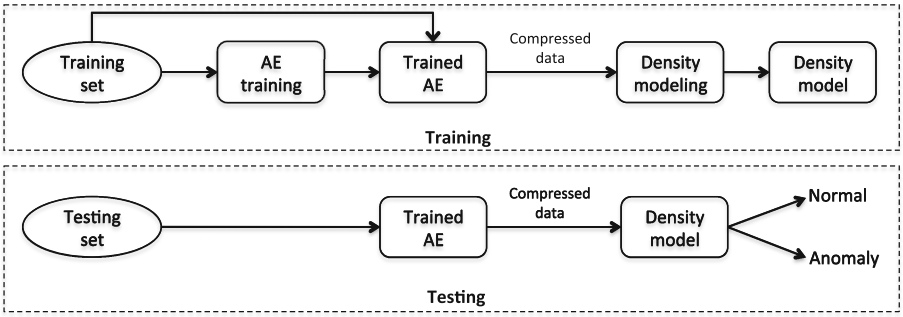


Fig. 2. The proposed anomaly detection model

5 Experiments

5.1 Datasets

The approach of simulating a one-class dataset by throwing away data from a binary dataset is a common approach in previous work [2,5,21]. In this work, we choose datasets that have one class considered as normal class and other classes treated as an anomaly class [4,21]. Four UCI datasets [14], namely Wisconsin Breast Cancer Database (WBC), Wisconsin Diagnostic Breast Cancer (WDBC), Cleveland heart disease (C-heart) and Australian Credit Approval (ACA), and NS-KDD dataset [20] are employed for our experiments. For the UCI datasets, we randomly sample 70 % for training and 30 % for testing. The normal training set is formed by removing all anomaly examples.

NSL-KDD dataset is a filtered version of the KDD Cup 1999 dataset [12] after removing all redundant instances and making the task more difficult. Each record in the dataset is labeled as either normal or as a specific kind of attack belonging to one of the four main categories: Denial of Service (DoS), Remote

to Local (R2L), User to Local (U2R) and Probe. NSL-KDD consists of two datasets: $KDDTrain^+$ and $KDDTest^+$ which are drawn from different distributions. Several of the variables in the dataset are categorical or discrete. We simply treat them as real-valued. As shown in Sect. 6 this gives good results, but better encodings are possible.

In this work, we plan to conduct our experiments on four groups of attacks separately. The aim is to see how efficiently our method performs on each group of attacks. In order to transfer hyperparameter values from the UCI to the NSL-KDD datasets, we wish to have a similar-sized dataset. We randomly subsample 350 normal instances from $KDDTrain^+$. We use all normal and anomaly instances from $KDDTest^+$ as our labelled test set. The details are shown in Table 1.

Table 1. One-class classification datasets

Dataset	Features	Training set	Testing set	
		Normal	Normal	Anomaly
C-heart	13	112	48	42
ACA	14	268	115	93
WBC	9	310	134	72
WDBC	30	249	108	64
DoS	41	350	9711	7458
R2L	41	350	9711	2887
U2R	41	350	9711	67
Probe	41	350	9711	2421

5.2 Experimental Settings

In this work, the classifiers will be constructed from the normal class only. There is no validation set for doing cross-validation. Therefore, we plan to conduct two experiments, one preliminary experiment for tuning the hyperparameters of the proposed models and one main experiment for evaluating the models. We use the terms OCCEN, OCKDE, and OCAE to refer to one-class classifiers based on the hybrid of AE and Centroid, the hybrid of AE and KDE, and AE itself respectively. The choice of a threshold for classifiers in practice varies from domain to domain, but in this work we try many different thresholds, and evaluate the area under the resulting ROC curve (AUC).

The parameters that will not be estimated from the preliminary experiment are set to common values. The Adaptive Gradient Algorithm (Adagrad) [6] with a common value for the learning rate, $\alpha = 0.01$, and smoothing term $\varepsilon = 10^{-8}$ will be used to train AEs. Hawkins et al. [9] chose different values for *epochs* (from 1000 to 40000), but in this work we choose a single value for $epochs = 5000$. The Gaussian kernel is used for KDE and its bandwidth, $h = \sqrt{\frac{hidden\ size}{2}}$ as in [16].

The preliminary experiment is done on the four UCI datasets to investigate steepness, k, and estimate the size of hidden layer, m, of the models for later testing on the NS-KDD dataset. Firstly, we visualise the data distribution in the hidden layer. In Fig. 3, normal and anomaly data are plotted with different values of k (0.1, 0.5, 1.0) and $m = 2$. We see normal data is approximately Gaussian for $k = 0.1$ whereas for $k = 1.0$ it seems to be distributed along the borders of a hyperbox. Even in this 2D example, for $k = 1.0$, the anomaly data is strongly concentrated in a single area, allowing good separation. We choose a common value $k = 1.0$ for our main experiment.

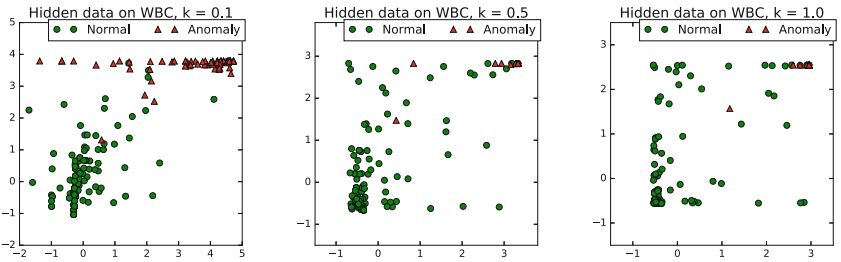


Fig. 3. Data on hidden layer with respect to hidden size, $m = 2$

Secondly, we run the models with different sizes of hidden layer on the four UCI datasets. In Fig. 4, the AUC from OCCEN, OCKDE and OCAE are plotted against hidden size. The figures illustrate that both the three classifiers produce very high AUC values at hidden size, $m = 4$ and 8 on WBC, and $m = 4$ on ACA. However, on C-heart, OCCEN and OCKDE perform very well at $m = 3$ and 6 whereas OCAE produces the highest AUC at $m = 4$. The highest and the second highest AUC values from the three classifiers on WBCD are obtained at $m = 8$ and 6 respectively. Overall, these classifiers produce good accuracy at $m = 3$ or 4 on WBC, ACA and C-heart, and at $m = 6$ or 8 on WBCD.

Therefore, we propose a rule of thumb for choosing hidden size, $m = [1 + \sqrt{n}]$, where n is the number of features. Based on this rule, we calculate parameter m for the models on these datasets, $m = 4$ for WBC, ACA and C-heart, and $m = 6$ for WBCD. For the NSL-KDD dataset, m will be equal to 7.

The main experiment is to investigate the performance of the methods OCAE, OCCEN, OCKDE on the four groups of attacks in NSL-KDD dataset. The classifiers are set up with the set of parameters presented above, and the results are shown in Table 2 and Fig. 5. The code of the experiments is available on github[1].

6 Results and Discussion

This section presents the experimental results of evaluating the proposed one-class classifiers on the four groups of attacks in NSL-KDD dataset. The performance of

[1] https://github.com/caovanloi/AEDensityEstimation.

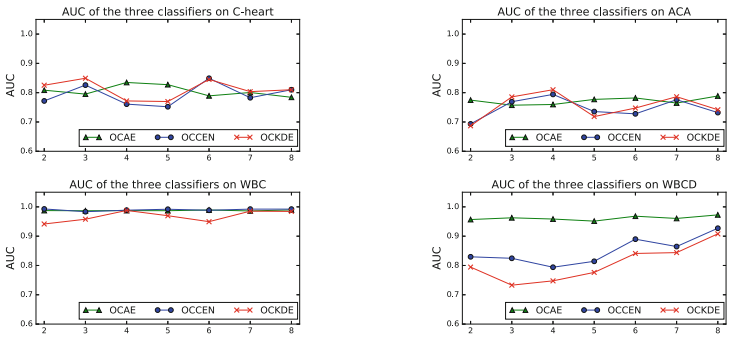


Fig. 4. Plotting AUC values against different hidden sizes on the UCI datasets

the three one-class classifiers, OCAE, OCCEN, OCKDE is evaluated using AUC. The results are summarized in Table 2. The ROC curves of the density-based classifiers are shown against those of OCAE in Fig. 5.

Table 2 illustrates the AUC values from the three one-class classifiers. It can be seen from the table that OCKDE performs very well in terms of accuracy, and better than OCAE on DoS, U2R and Probe. The AUC values from OCCEN are also higher than those from OCAE on Probe. However, the performance of OCCEN is similar to or worse than that of OCAE on three other groups.

The ROC curves are displayed in Fig. 5. The ROC curves of OCCEN, OCKDE are plotted against the ROC curve of OCAE. It can be seen that the curves of OCKDE is usually higher than the curves of the two other classifiers on the NSL-KDD dataset.

Table 2. The AUC results from the three classifiers on NSL-KDD dataset

Dataset	RE	AUC		
		OCAE	OCCEN	OCKDE
DoS	0.459	0.960	0.956	0.974
R2L	0.459	0.909	0.839	0.891
U2R	0.459	0.928	0.888	0.945
Probe	0.459	0.971	0.986	0.987

Overall, these results suggest that the proposed density-based classifiers, OCCEN and OCKDE, tend to perform well in terms of accuracy on datasets in which the normal and anomaly classes are highly separated (e.g. Probe, DoS or U2R). The KDE-based one-class classifier, OCKDE, is more powerful than OCCEN and OCAE in detecting anomalies from NSL-KDD dataset.

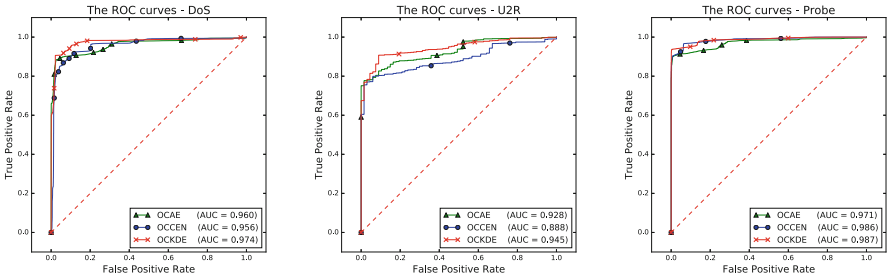


Fig. 5. The ROC curves of the three classifiers on NSL-KDD dataset

7 Conclusion and Further Work

In this paper, we have proposed a novel method for anomaly detection based on estimating density in the compressed hidden-layer representation of autoencoders. We have motivated this method through visualization of density in the hidden layer. We investigated the hyperparameters of AE based on the UCI datasets, and proposed an equation for estimating the size of hidden layer for later evaluating the models on NSL-KDD dataset.

The experimental results suggest that our proposed model performs well, and often out-performs a typical autoencoder approach based on reconstruction error on the dataset from security domain. The model also tend to work efficiently on the datasets in which the normal and anomaly classes are highly separated. This will help our model become an universal method for anomaly detection. Further work will focus on how to speed up the query stage of the models.

Acknowledgements. This work is funded by Vietnam International Education Development (VIED) and by agreement with the Irish Universities Association.

References

1. Aggarwal, C.C.: Outlier Analysis. Springer Science & Business Media, Berlin (2013)
2. Cao, V.L., Nicolau, M., McDermott, J.: One-class classification for anomaly detectionwith kernel density estimation and genetic programming. In: Heywood, M.I., McDermott, J., Castelli, M., Costa, E., Sim, K. (eds.) EuroGP 2016. LNCS, vol. 9594, pp. 3–18. Springer, Berlin (2016)
3. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. (CSUR) **41**(3), 15 (2009)
4. Curry, R., Heywood, M.: One-class learning with multi-objective genetic programming. In: IEEE International Conference on Systems, Man and Cybernetics, ISIC, pp. 1938–1945. IEEE (2007)
5. Curry, R., Heywood, M.I.: One-class genetic programming. In: Vanneschi, L., Gustafson, S., Moraglio, A., De Falco, I., Ebner, M. (eds.) EuroGP 2009. LNCS, vol. 5481, pp. 1–12. Springer, Heidelberg (2009)
6. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. **12**, 2121–2159 (2011)

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 Recogn. **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. Hawkins, S., He, H., Williams, G.J., Baxter, R.A.: Outlier detection using replicator neural networks. In: Kambayashi, Y., Winiwarter, W., Arikawa, M. (eds.) DaWaK 2002. LNCS, vol. 2454, pp. 170–180. Springer, Heidelberg (2002)
10. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)
11. Japkowicz, N., Myers, C., Gluck, M., et al.: A novelty detection approach to classification. In: IJCAI, pp. 518–523 (1995)
12. KDD Cup Dataset (1999). http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
13. Lee, W., Stolfo, S.J., Mok, K.W.: A data mining framework for building intrusion detection models. In: Proceedings of the 1999 IEEE Symposium on Security and Privacy, pp. 120–132. IEEE (1999)
14. Lichman, M.: UCI Machine Learning Repository (2013). http://archive.ics.uci.edu/ml
15. Moya, M.M., Koch, M.W., Hostetler, L.D.: One-class classifier networks for target recognition applications. Technical report, Sandia National Labs., Albuquerque, NM (United States) (1993)
16. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
17. Sakurada, M., Yairi, T.: Anomaly detection using autoencoders with nonlinear dimensionality reduction. In: Proceedings of MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, p. 4. ACM (2014)
18. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443–1471 (2001)
19. Shafi, K., Abbass, H.A.: Evaluation of an adaptive genetic-based signature extraction system for network intrusion detection. Pattern Anal. Appl. **16**(4), 549–566 (2013)
20. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: NSL-KDD Dataset (2009). http://www.unb.ca/research/iscx/dataset/iscx-NSL-KDD-dataset.html
21. To, C., Elati, M.: A parallel genetic programming for single class classification. In: Proceedings of 15th Annual Conference Companion on Genetic and Evolutionary Computation, pp. 1579–1586. ACM (2013)
22. Veeramachaneni, K., Arnaldo, I., Cuesta-Infante, A., Korrapati, V., Bassias, C., Li, K.: AI^2: training a big data machine to defend. In: International Conference on Big Data Security. IEEE, New York (2016)