Neural Computing and Applications (2022) 34:20491–20505
https://doi.org/10.1007/s00521-022-07506-9

S.I. : CYBERSECURITY APPLICATIONS OF COMPUTATIONAL INTELLIGENCE

Efficient anomaly detection through surrogate neural networks

Martin Flusser[1,2] · Petr Somol[3]

Received: 3 March 2021 / Accepted: 1 June 2022 / Published online: 30 June 2022

Abstract

Anomaly Detection can be viewed as an open problem despite the growing plethora of known anomaly detection techniques. The applicability of various anomaly detectors can vary depending on the application area and problem settings. Especially in the Big Data industrial setting, an important problem is inference speed, which may render even a highly accurate anomaly detector useless. In this paper, we propose to address this problem by training a surrogate neural network based on an auxiliary training set approximating the source anomaly detector output. We show that existing anomaly detectors can be approximated with high accuracy and with application-enabling inference speed. We compare our approach to a number of state-of-the-art algorithms: one class k-nearest-neighbors (kNN), local outlier factor, isolation forest, auto-encoder and two types of generative adversarial networks. We perform this comparison in the context of an important problem in cyber-security—the discovery of outlying (and thus suspicious) events in large-scale computer network traffic. Our results show that the proposed approach can successfully replace the most accurate but prohibitively slow kNN. Moreover, we observe that the surrogate neural network may even improve the kNN accuracy. Finally, we discuss various implications that the proposed approach can have while reducing the complexity of applied anomaly detection systems.

Keywords Anomaly detector · Neural network · Model transfer · Detector ensemble · High-performance anomaly detection

1 Introduction

Anomaly detection (AD) is gaining on importance with the massive increase of data we can observe in every domain of human activity. In many applications, the goal is to recognize objects or events of classes with unclear definitions and missing prior ground truth, while the only assumed certainty is that these entities should be different from what we know well. The problem can thus be seen as the problem of modeling what is common and then identifying outliers. Anomaly detection is a crucial technique in cyber-security, industrial quality control, banking, credit card fraud detection, medical diagnostics and many other fields [13].

Although AD as a general problem has been widely studied (cf. Sect. 2), progress is arguably slower than in supervised learning. Particularly, the recent rapid advances in neural networks for classification (see, e.g., [16, 24]) seem harder to replicate in AD. The primary neural models used in AD are unsupervised generative models, typically auto-encoders (AE) or generative adversarial networks (GAN) [12]. Although there is great promise in GAN models [1], they can be more difficult to successfully apply [52] than traditional techniques. Traditional techniques thus often remain the straightforward choice, especially in industrial applications. Among traditional AD principles, density-based techniques like k-nearest neighbor (kNN) [7, 28], isolation forest [36] or local outlier factor [9] quite often achieve surprisingly good accuracy. At the same time, many such models can become computationally expensive or even prohibitive in an industrial setting.

We would like to thank Jan Brabec for consultations and for sharing expertise in the field. This work has been supported by the Grant Agency of the Czech Technical University in Prague, grant No.SGS20/188/OHK4/3T/14.

✉ Martin Flusser
flussmar@fjfi.cvut.cz

Petr Somol
somol@utia.cas.cz

[1] Faculty of Nuclear Sciences and Physical Engineering, Czech Technical University in Prague, Prague, Czechia

[2] Cognitive Intelligence, Cisco, Prague, Czechia

[3] Institute of Information Theory and Automation, Czech Academy of Sciences, Prague, Czechia

To overcome this problem, we would need either to reduce the complexity of an existing AD without compromising its accuracy, or to approximate it by a different, cheaper, but comparably accurate model. Although indirectly related ideas exist (e.g., [23]), there seems to be a lack of solutions addressing this problem in the AD context. For that reason, we proposed to address the problem using neural networks due to their efficient inference speed and their mature support in an industrial setting [39]. In [20], we have shown that for an existing *k*NN anomaly detector a surrogate density-approximating neural network model can be indeed constructed with comparable accuracy and higher inference speed. The basic idea is simple: First, a set of generated auxiliary samples is constructed with each sample labeled by its anomaly score as inferred by *k*NN. Second, a multilayer perceptron (MLP) neural network is trained from such auxiliary data. (For visual illustration on a two-dimensional projection of benchmark data set Abalone [18] with $k = 5$, see Figs. 1, 2 and 4.)

In this paper, we extend the idea and apply it to a large-scale problem in cyber-security. We investigate a richer set of parametrization options and evaluate the impact of parameters in all stages of model construction. Also, we provide a comparison of surrogate models to multiple state-of-the-art anomaly detectors. We show that surrogate models can effectively approximate a well-performing AD in the industrial setting and thus provide a model with higher inference speed and/or lower memory footprint. The success of a surrogate model is, however, data and source AD dependent.

The paper is structured as follows: In Sect. 2, we review existing AD methodology, in Sect. 3 we introduce the concept of surrogate neural networks, in Sect. 4 we cover the experimental evaluation of the proposed methodology, in Sect. 4.4 we discuss its robustness, in Sect. 5 we discuss additional application options, and in Sect. 6, we provide a summary and conclusion.

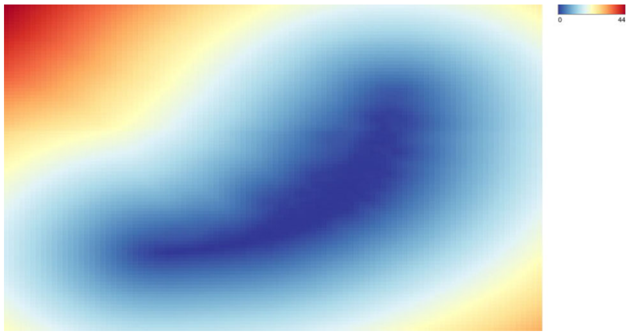

Fig. 1 Heat map illustration of anomaly scores induced by *k*NN anomaly detector on benchmark Abalone data set

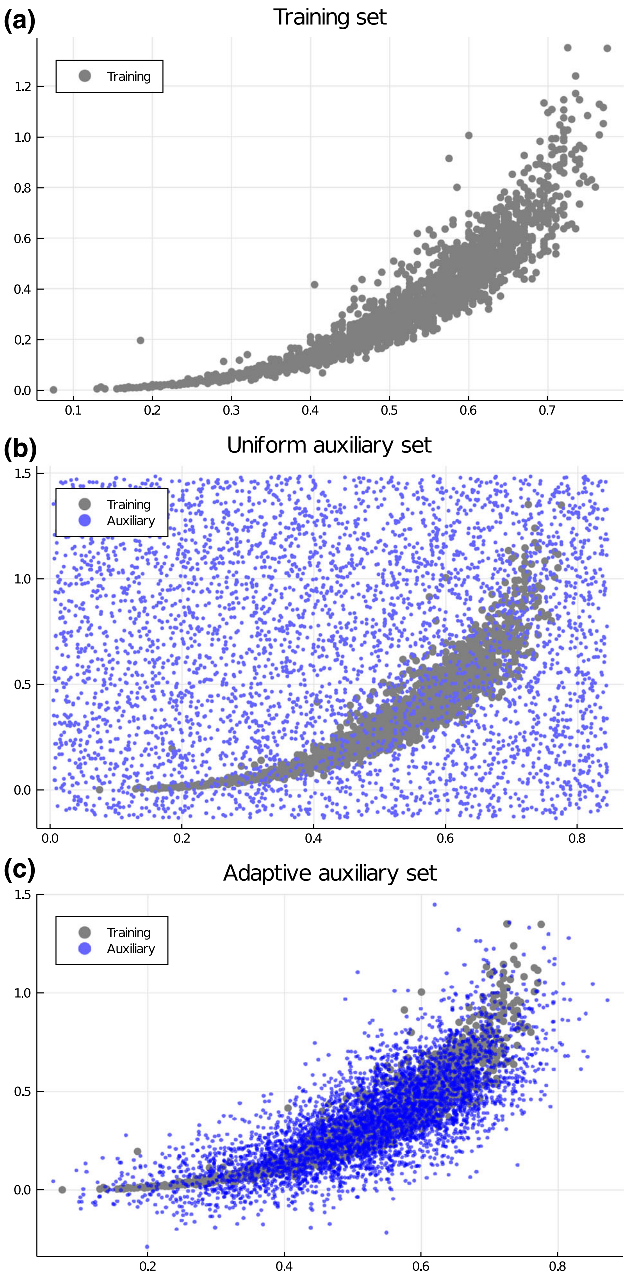

Fig. 2 Following the example in Fig. 1 we construct auxiliary set(s) to be eventually used to train a neural anomaly detector. Auxiliary samples need to cover all areas of notable *k*NN anomaly score variance. See **a** input data (gray dots), **b** uniform auxiliary set (blue dots overlaid over input data), **c** adaptive auxiliary set (blue dots overlaid over input data). Labeling of auxiliary samples is illustrated in Fig. 4

2 Prior art

There are a number of methods for anomaly detection the survey of which is given, e.g., in [13] or [12].

Nearest neighbor techniques [5] are popular due to their simplicity, reliable accuracy (which is often unsurpassed, cf. [52]) and adaptability to various data types. Their computational complexity, however, grows rapidly with both the dimensionality and size of the training data. Supporting structures thus have been proposed. The *k-d tree* [6, 8, 22] is a binary search tree that uses hyperplanes to divide the space to accelerate the search. The *ball tree* [8, 56] uses hyperspheres to cover the space recursively. Despite these advances, the problem of *k*NN computational complexity cannot be considered as resolved.

Local outlier factor (LOF) [9] and subsequent ideas like probabilistic LOF [35] can be useful with unevenly distributed data sets. The key idea here is to deem a sample anomalous if it is significantly farther from its neighbors than they are from each other.

Isolation forest (IF) [36] and subsequent ideas like *extended* [29], *functional* [53] or *kernel* isolation [55] proved to be practical for high-dimensional problems. The key idea is to build projection trees and evaluate at which depth a sample becomes isolated. It is expected that more anomalous samples are easier to isolate, thus appearing closer to the tree root.

AD can be naturally performed using statistical approximation models. *Gaussian mixture models* have been shown useful in mammography [27]. A simple option is to utilize *Parzen* models [60].

The standard anomaly detection knowledge base also includes *kernel PCA* methods [38], *kernel density estimation* (KDE) including *robust* KDE [32] and *one-class support vector machines* (SVM) [49] that all have been compared to and partly outperformed by neural models, see, e.g., [62].

The simplest form of a neural network traditionally used for unsupervised AD is *auto-encoder* [2], where reconstruction error typically serves as a proxy to measure the anomality. Many extensions of the idea exist, e.g., [4, 15, 46, 58, 59, 62]. Auto-encoders can be viewed as the primary unsupervised neural AD technique. They do not model the distribution of anomalies, but optimize a proxy criterion like the reconstruction error. This can limit the success of AEs.

Other types of neural network AD models depend on additional knowledge about possible outliers or other indirect information about the anomaly (see, e.g., [11, 40, 45, 47]).

More recently, various types of *generative neural models* have been applied to anomaly detection in fields including clinical imaging, industrial time series and intrusion detection [31, 48, 61]. A particularly successful semi-supervised *GANomaly* model has been applied in X-ray screening [1]. The model jointly learns the generation of high-dimensional image space and the inference of latent space. For a wider overview of generative AD techniques, see [12].

Other types of neural networks have been used to estimate and simulate the nature of the anomalies in the training phase. In other words, *auxiliary data* is used (explicitly or implicitly) when training the neural model. An intuitive auxiliary set with binary labels was utilized in [25] to represent the manifold. However, the method is limited by the assumption that the non-anomalous data lie on a well-sampled, locally linear low-dimensional manifold. The auxiliary set consists of the training set and potentially anomalous samples that are generated with the Euclidean radius around the training data. The authors claim that the collision probability of the generated anomalous samples and training set is low due to the assumptions.

Supervised AD is performed in [30] where *outlier exposure* is used to train the model with an auxiliary set skimmed from other sources (i.e., pictures skimmed from the web) in addition to the training set. The auxiliary set is purified not to contain similar samples to the training class, thus, the detector is effectively trained with two different classes. The authors also consider the difficulty of creating the artificial auxiliary set (i.e., with Gaussian noise) that teaches the network to generalize the unseen anomaly distributions in the unsupervised scenario in contrast to the supervised.

The ideas investigated in this paper follow an alternative approach to unsupervised AD consisting of neural network-based approximation of an existing anomaly detector's score function. Initially introduced in our previous works [20, 21], this class of methods depends on auxiliary samples with a non-binary label, which cover both the area of the training set (non-anomalous) and its close and more distant neighborhood, i.e., the area potentially significant to detect anomalies.

In many applications, it has been shown that *ensembles of anomaly detectors* perform better than a single detector [14, 50, 54, 64]. This is common, particularly in cybersecurity [26, 57].

In the rest of this paper, we will focus on training surrogate models mainly for nearest neighbor anomaly detectors. We will then provide a comparison to various state-of-the-art anomaly detectors listed above, including the density-based and generative neural model-based detectors.

3 Surrogate neural networks for anomaly detection

The methodology we propose aims at approximating an existing anomaly detector's score function using a surrogate neural network. Let us refer to the detector to be approximated as the *source anomaly detector*.

First, we create an auxiliary data set covering the source detector's input space. For each sample in the auxiliary data set, the source detector's anomaly score is computed and assigned to the sample as its label. Then, the auxiliary data set is used to train a standard multilayer perceptron (MLP) with a single output.

Having the training set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ where $\mathbf{x}_i \in \mathbb{R}^d, \forall i \in \{1, \ldots, n\}$, and $\mathbb{R}^d$ is a d-dimensional vector space. Let us denote $\mathbf{A}$ the auxiliary data set of m samples where

$$\mathbf{A} = \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m\},\ \mathbf{a}_i \in \mathbb{R}^d,\ \forall i \in \{1, \ldots, m\}$$

and Y be the vector of respective anomaly scores computed using the source anomaly detector, where $Y = \{y_1, y_2, \ldots, y_m\}$, $y_i \in \mathbb{R}$, $\forall i \in \{1, \ldots, m\}$.

For simplicity, we assume that the *size* of MLP hidden layers and their *number* can be viewed as hyper-parameters p and q, respectively, and that both parameters can be determined through hyper-parameter search. In Sect. 4.2.2, we discuss details of the search for our specific cybersecurity problem.

3.1 Auxiliary data sets

At first, the auxiliary set $\mathbf{A}$ needs to be computed from the training set $\mathbf{X}$. The actual auxiliary set construction can be done in many ways. In the following, we discuss two options: the trivial coverage of the input space by a hyper-block and efficient construction, which employs the distribution of the input data. Figure 2 illustrates the two options.

3.1.1 Uniform auxiliary data set

The idea of the *uniform auxiliary set* construction from [20] is naïve as it attempts to cover the space uniformly on a rectangular subspace defined as the smallest enclosing hyper-block that contains all points in the input data space. More specifically:

1. A bounding hyper-block of $\mathbf{X}$ is determined as the smallest enclosure of the input data, defined by the vector of lower bounds $\mathbf{h}_l$ and upper bounds $\mathbf{h}_u$ such that $\mathbf{h}_l^{(j)} \leqslant \mathbf{x}_i^{(j)} \leqslant \mathbf{h}_u^{(j)} \quad \forall i \in \{1, \ldots, n\} \quad \forall j \in \{1, \ldots, d\}$ where $\mathbf{x}_i^{(j)}$ represents jth element of ith vector from $\mathbf{X}$
2. The hyper-block is filled with randomly generated and uniformly distributed samples $\{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m\}$. By default we consider uniform random sampling. Note that the choice of m for concrete problem may depend on n and d (see also Sect. 4.2.2).
3. The anomaly score vector Y is constructed so that for each auxiliary sample $\mathbf{a}_i, i \in \{1, \ldots, m\}$ the respective $y_i \in Y$ is computed as the source anomaly detector's score on $\mathbf{a}_i$.

Remark: in case of kNN the y_i is computed as mean distance $G(\cdot)$:

$$y_i = G(\mathbf{a}_i) = \frac{1}{k}\sum_{j=1}^{k} D_j(\mathbf{a}_i) \qquad (1)$$

where $D_j(\mathbf{a}_i)$ represents the jth smallest distance of $\mathbf{a}_i$ to samples from $\mathbf{X}$. Note that the number of neighbors k is a parameter [37, 63].

3.1.2 Adaptive auxiliary data set

The uniform auxiliary set as defined above is sub-optimal due to multiple reasons. Clearly, the distribution of points in the uniform auxiliary set does not reflect the varying importance of various regions in the auxiliary space; the uniform auxiliary set can easily waste sampled points in regions of no importance while lacking coverage in dense and complicated manifolds.

This problem gets worse with increasing dimensionality. This "curse of dimensionality effect" can be illustrated by the simple example of data distributed within a hyper-sphere of unit radius. Assuming we have the hyper-sphere enclosed in an auxiliary hypercube, the ratio of hyper-sphere volume over hypercube volume decreases with increasing dimensionality (see Fig. 3). Only a negligible fraction of auxiliary samples would be relevant in problems with more than low single-digit dimensionality.

Another problem with hyper-block is the possible loss of information. Auxiliary data generated strictly within a hyper-block cannot approximate the continuity of anomaly scores with growing distance from the input samples. The

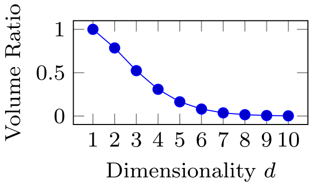

Fig. 3 Inefficiency of covering input space by uniform auxiliary data sets: d-dimensional hyper-sphere to hypercube volume ratio

sharp auxiliary set boundary thus can distort the eventual surrogate model.

To resolve both of the problems above, we propose to construct the auxiliary data set adaptively to reflect the distribution in input data. This is achieved by generating auxiliary samples according to a modified Parzen estimate of the input density. No bounding hyper-block is thus needed, while the auxiliary samples now become more frequent in areas of more detail.

Such auxiliary data set should provide more detailed coverage of anomaly score distribution than the uniform auxiliary data set with the same number of auxiliary samples. The *adaptive auxiliary data set* is constructed as follows:

1. Optimal variance h for Parzen window approximation of $\mathbf{X}$ is determined (by default we use cross-validation and random search on training data).
2. The auxiliary set $\mathbf{A} = \{\mathbf{a_1}, \mathbf{a}_2, \ldots, \mathbf{a}_m\}$ is generated as realization of the Parzen distribution as follows: iterate over samples of $\mathbf{X}$ and create $\mathbf{a_i} = \mathbf{x_i} + \mathcal{N}(0, h \cdot k_{var})$ where k_{var} (variance multiplicative coefficient) is a parameter:

   $$\forall i \in \{1, \ldots, m\} : \ \mathbf{a_i} = \mathbf{x}_{(\mathbf{i} \bmod \mathbf{n})} + \mathcal{N}(0, h \cdot k_{var})$$

   Note that if $m > n$, multiple auxiliary samples get generated based on a single input sample. The choice of m and k_{var} for the concrete problem is discussed in Sect. 4.2.2.
3. The anomaly score vector Y is constructed in the same way as for uniform auxiliary set (see Sect. 3.1.1, step 3).

See Fig. 2 for the difference between *uniform* and *adaptive* auxiliary sample distributions. See Fig. 4 for the same auxiliary sample distributions enriched by anomaly score labels. The impact of the improved *adaptive* auxiliary set efficiency is also shown in Fig. 11.

3.2 Training the surrogate neural model

We can now train a multilayer perceptron (MLP) on the auxiliary training set $\mathbf{A}$ to predict anomaly scores Y. We parametrize the size of hidden layers p and the number of hidden layers q (see Fig. 5). We minimize the mean squared error (MSE) between the *predicted* scores and *ground truth* scores using the *Adam* optimizer [33]. For further details of our experimental setup, see Sect. 4.2.2.

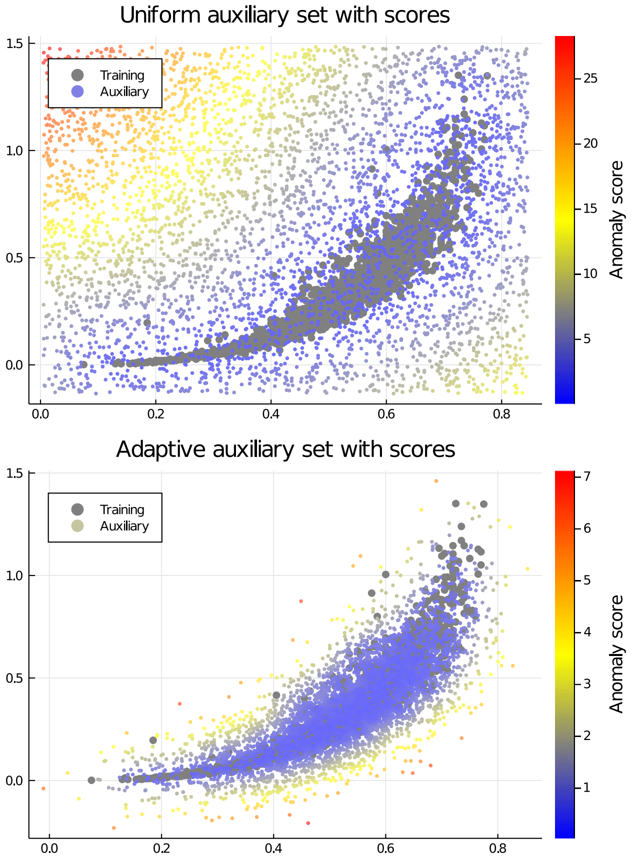

Fig. 4 Following the example in Figs. 1 and 2, we finalize the construction of training set(s) to be eventually used to train a neural anomaly detector. We use the auxiliary set(s) that cover the space of notable kNN anomaly score variance as shown in Fig. 2. Each auxiliary sample gets labeled by its respective anomaly score (illustrated here by colors on a heat scale) computed using kNN from input data (overlaid gray dots). Compare the distribution of auxiliary anomaly scores to the heat map in Fig. 1

4 Experimental evaluation

We evaluate the proposed approach on a real problem in cyber-security—the discovery of outlying (and thus suspicious) events in large-scale computer networks. For this purpose, we use network traffic telemetry data.

To obtain a baseline, we evaluate a number of state-of-the-art algorithms: one class kNN [5], LOF [9], IF [36], auto-encoder, and generative adversarial networks [1, 12] as well as a simple Parzen detector [60]

Subsequently, we construct a surrogate anomaly detector from the best performing (but slow) baseline source detector. We then include the surrogate detector in the overall comparison. We primarily verify the achieved improvement of inference speed. Secondarily, we verify

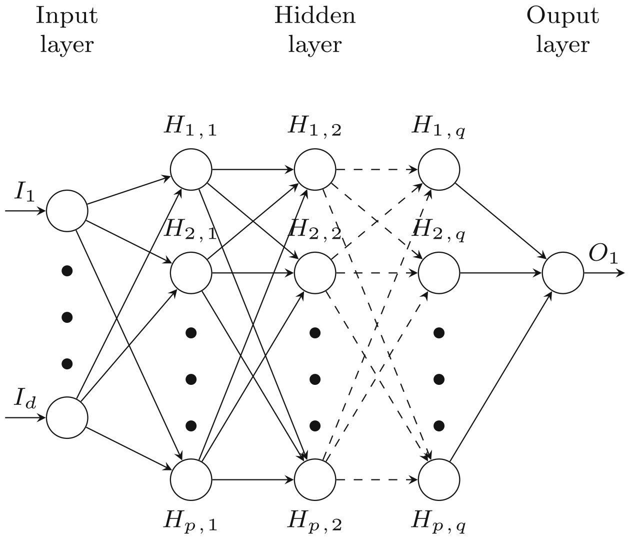

Fig. 5 Default structure of the surrogate neural network. The size of hidden layers p and the number of hidden layers q are problem-dependent parameters subject to optimization

whether the surrogate detector can match the accuracy of the baseline source detector.

Additionally, we test the robustness of the proposed surrogate approach to variations in auxiliary data set parametrization and to modifications of the auxiliary set construction procedure.

4.1 Cyber-security data set

We perform the evaluation on a data set provided by Cisco Systems, described in detail in [34]. The data represents persistent connections observed in computer network traffic using the *NetFlow* protocol. A connection between each device–server pair is logged as a series of flow records. Because a single flow record contains only minimal usable information (transferred data sizes, timing and source–destination identifiers), it is needed to build models from at least sequences of flows. We follow the methodology from [34] where a series of flows is transformed into a vector through feature extraction. The features are expert-designed with the aim to maximally preserve connection characteristics. They are:

- Average flow duration
- Flows inter-arrival times mean
- Flows inter-arrival times variance
- Target autonomous system uniqueness
- Target autonomous system per-service uniqueness
- Unique local ports count
- Byte count weighted by target autonomous system uniqueness
- Device overall daily activity deviation from normal
- Remote service entropy
- Remote service ratio

In our data set, each sample vector represents a 5-minute traffic window, which is the standard in industrial detection systems. The number of samples is 222 455. The data is multi-class, with classes distinguishing various types of benign and malicious connections.

4.1.1 Anomaly detection levels of difficulty

In order to evaluate anomaly detectors on the available data, we adopt the experimental protocol of Emmott [19]. The protocol defines the transformation of multi-class input data into AD benchmark data. To give a varied view of the evaluated anomaly detectors, we have used the Emmott protocol to produce four AD benchmark data sets of increasing difficulty. Emmott's procedure categorizes malicious classes into four groups based on the evaluation of the anomalousness level of the respective malicious samples. Then, for each of the four groups, a new data set is constructed, taking all benign samples together with samples from the single respective group. In the following we will thus refer to *easy*, *medium*, *hard* and *very hard* problems.

4.2 Evaluation setup

To construct the training and testing sets, random resampling (8×) is adopted such that for each sampling iteration, 75% of normal (non-anomalous) samples are utilized for training while the remaining 25% are utilized for testing. The anomalous samples are used only in the testing phase. Anomaly detectors' inference speed is measured in seconds on a single Intel Core i7 vPro 8th Generation. Detection accuracy is measured with AUC of ROC [10] as is common in the literature. Remark: we choose random resampling over cross-validation consistently with the literature [17, 42, 51] to mitigate the data imbalance problem in AD testing.

4.2.1 Setup of baseline detectors

For the evaluation, we set the parameters and use the baseline anomaly detectors as follows.

To evaluate kNN accuracy, we compute AUC according to the anomaly score obtained as mean distance $G(\cdot)$ introduced in Eq. (1). The optimal choice of the parameter k which is essential for kNN is not addressed in this paper. However, we observed $k = 5$ as the best performing across our experiments. To evaluate kNN inference speed, we need to take into account the important kNN variants optimized for speed. Therefore, we evaluate the *basic* kNN which is implemented as a brute tree, *k-d tree* and *ball tree* which both implement supporting structures for faster nearest neighbor search (see Sect. 2). The usage of supporting structures does not affect accuracy. However, the inference speed may differ significantly.

For *isolation forest*, we performed a grid search to choose the best number of trees from $\{100, 200, 400\}$ and the number of samples from $\{256, 512\}$. We select the best performing parameters (on validation data) for each problem difficulty.

For *local outlier factor*, we set the parameter $k = 5$.

For *auto-encoder* evaluation, we opt for the de-noising three-layer AE according to [15, 58]. When computing AUC, the anomaly score is proxied by reconstruction error. AEs are subject to parametrization. We performed a meta-optimization procedure to choose the type and magnitude of noise and the number of neurons per hidden layer. We observed that *Gaussian noise* worked better than *salt and pepper noise*. Four different magnitudes of noise have been tried with deviations between 0.01 and 0.2 while the samples were scaled to [0, 1] for each dimension. The number of hidden neurons was selected with a full-grid search in $\{1, \ldots, 10\}$. We observed only negligible improvement from repeated random initialization. All models were trained in 300 000 iterations; varying the number of iterations also proved to have only negligible impact. The eventual meta-optimization procedure consists of building the 40 models (4 noise parameters, 10 hidden layer size parameters) and choosing the one with the best achieved AUC on validation data for each problem difficulty.

We include two forms of *generative adversarial networks* in the evaluation. First, we include the GAN-based AD by Zenati et al. [61]. Specifically, we used the implementation from [51] and optimized the respective parameters on the following ranges: *dim*(z) on $\{2, 4, \ldots, 256\}$, *number of dense layers* on $\{2, 3, 4\}$, and α on $\{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$. See [51] for details of the parameters and their recommended ranges.

To evaluate the *GANomaly* AD [1], we also used the implementation and from [51]. We optimized the respective parameters on the following ranges: *decay* on $\{0, 0.1, \ldots, 0.5\}$, w_{adv}, w_{con}, w_{enc} on $\{1, 10, 20, \ldots, 100\}$, λ on $\{0.1, 0.2, \ldots, 0.9\}$, $R(x)$ and $L(X)$ on $\{\text{MAE, MSE}\}$, *number of convolution layers* on $\{1, 2, 3, 4\}$ and *number of channels* $\{8, 16, \ldots, 128\}$. See [51] for details of the parametrization.

For the *Parzen*-based AD, we use the same setup as described in Sect. 4.4.2. The Gaussian kernel is optimized with cross-validation on the training data. The parameter k_{var} is selected from $\{1, 2, 3, \ldots, 10\}$ for best performance on validation data.

4.2.2 Surrogate neural network setup

Based on the observation that *k*NN achieves outstanding accuracy but poor inference speed on our cyber-security problem, we chose it as the source anomaly detector for building the *surrogate neural network*. We parametrize the surrogate model as follows.

We fixed $k = 5$ in *k*NN used for auxiliary data set construction to get results comparable to the standalone *k*NN baseline anomaly detector. The auxiliary data set is constructed as described in Sect. 3.1 with the total number of auxiliary samples set to $m = n \cdot d$. (We discuss the impact of auxiliary set parametrization further in Sect. 4.4.)

ReLU activation function is used for all neurons (except for the input ones). The size of the batch is set always to 80. We use MSE as the loss function and train the network with *Adam* optimizer.

We opted for a simple meta-optimization of neural model parameters. For each problem difficulty (see Sect. 4.1.1) we train multiple models, to eventually retain the one with the best loss on validation data. The variation across training runs consists in: the number of hidden layers q varies between values $\{1, 2, 3\}$, hidden layer size p varies between values $\{1d, 3d, 5d, 7d, 9d\}$, random weight initialization is repeated 4×, the number of iterations is thresholded by six values between 15000 and 700000.

Figure 6 illustrates the impact of parameters q and p on the achieved accuracy.

4.3 Results

While addressing the cyber-security problem, our primary concern is the ability of surrogate models to improve the inference speed of the best performing baseline anomaly detector. Our secondary concern is the ability of a surrogate model to match the accuracy of its source anomaly detector.

4.3.1 Inference speed

We evaluated the inference speed of all baseline anomaly detectors and compare it to the speed of the surrogate detectors. Inference speed can notably depend on the problem dimensionality. The speed of some detectors—especially the nearest neighbor-based ones—also strongly depends on training data size. We illustrate this observation in Fig. 7. All measurements have been done on *medium* problem difficulty. Graphs show the time needed to process all samples in the test set, i.e., 25% of all available data.

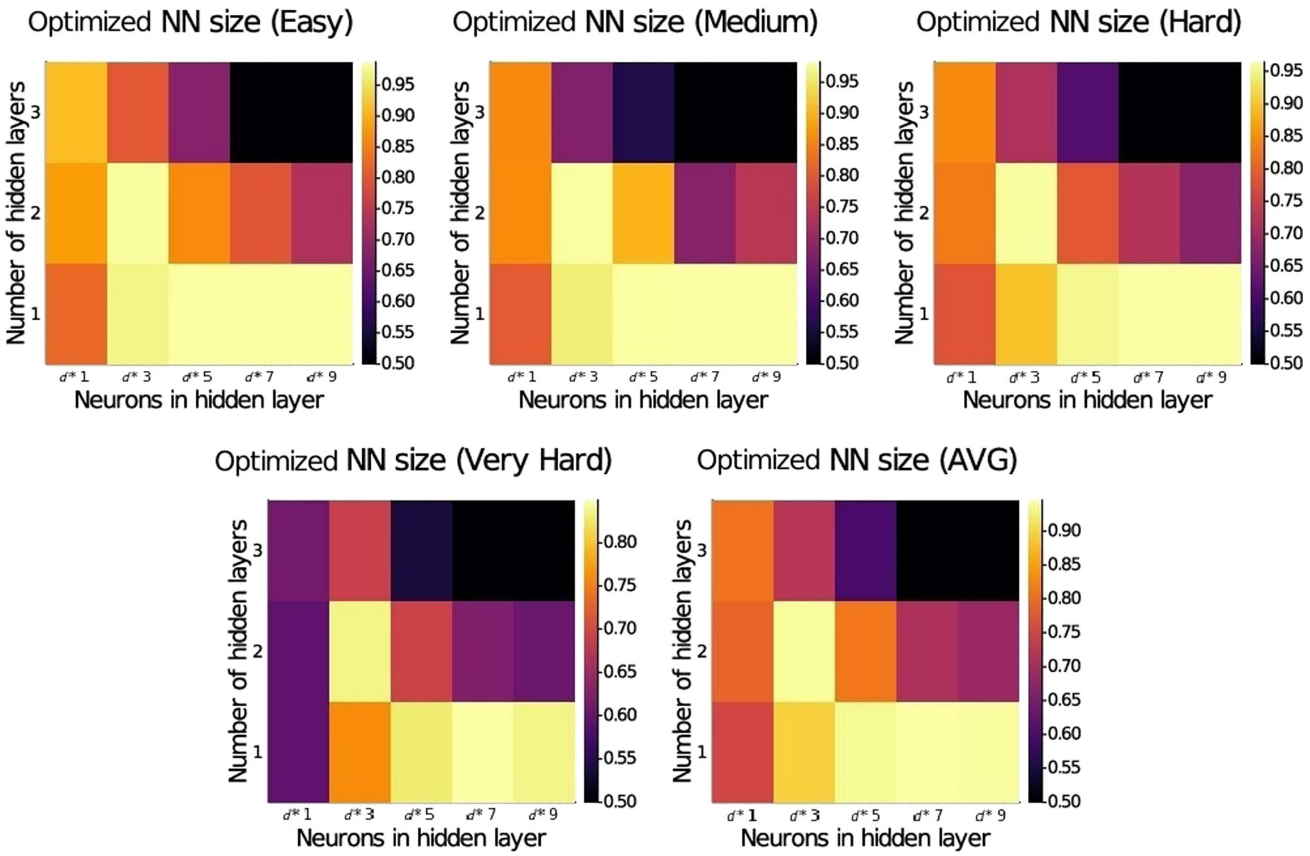

Fig. 6 Illustrating the impact of surrogate neural network architecture to detection accuracy for the four different problem difficulties: *easy* , *medium* , *hard* and *very hard* (the 5th diagram shows the average). Brighter color depicts higher achieved AUC of ROC

Note that the graphs depict inference speed only, the training overhead is not included. The top plots show inference speed with respect to the size of the training set, the bottom plots show inference speed with respect to the dimensionality. The left plots are shown in linear scale, the right plots are shown in logarithmic scale.

As shown in Fig. 7, the inference speed of the neural network-based detectors is the least dependent on the size of training data. Dependence on the size of training data is most notable with nearest neighbor-based detectors, although the supporting structures in ball trees and k-d trees reduce this problem notably. Remark: in computer network analytics the number of samples to be processed online is several orders of magnitude higher than illustrated here.

The neural models including the proposed surrogate model in this test perform faster than the fastest nearest neighbor anomaly detector by at least an order of magnitude. Graphs suggest that this advantage will continue to grow with increasing dimensionality of the problem and even more so with growing training data set size.

Remark: Missing values in Fig. 7 are due to the limitation of the GAN implementation from [61].

4.3.2 Accuracy

The best baseline accuracy on our cyber-security problem has been obtained from kNN anomaly detectors on all problem difficulties with $k = 5$, reaching the average AUC of 0.945. Accordingly, we focused on constructing and evaluating surrogate neural network detectors with 5NN as the source anomaly detector. In Fig. 8, we compare the source and surrogate detectors visually using a 2D projection with anomaly score heat map (projection to the first two PCA principal components).

When evaluating the accuracy of surrogate anomaly detector models we primarily aim at verifying whether the surrogate model succeeds in matching the accuracy of its source anomaly detector. Improvement of accuracy is not expected although it can happen.

In Table 1, we primarily focus on comparing the surrogate neural network detectors (built from 5NN source detector with either uniform or adaptive auxiliary set) to the baseline 5NN detector. We then compare these to all the other baseline anomaly detectors. Each column in the table covers one problem difficulty, with the last column covering the average. Best achieved results are set in bold.

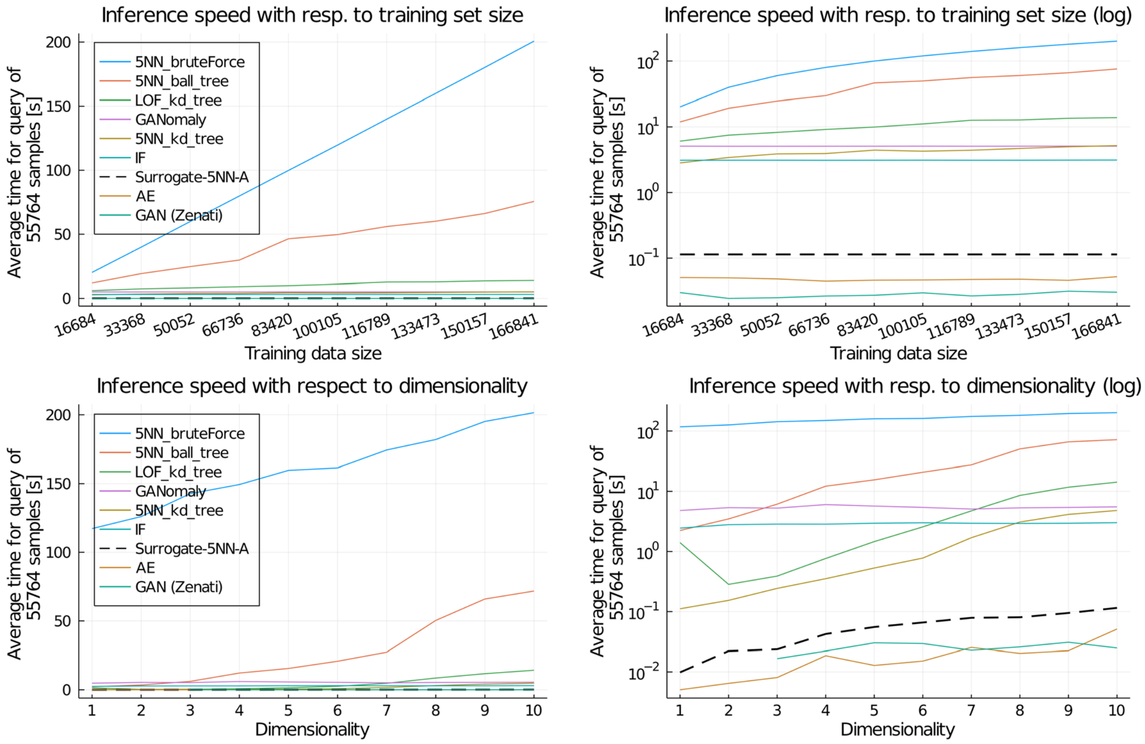

Fig. 7 Dependence of anomaly detectors' inference speed on *training data size* (top) and *dimensionality* (bottom) in application phase. Note the speed-up achieved through the surrogate neural network with the auxiliary set (dashed line) over its source anomaly detector 5NN. Graphs in linear scale (left) and logarithmic scale (right)

Assessing the accuracy of results over multiple data sets (difficulties) can be done in multiple ways [17]. We provide more detailed results in the form of confidence intervals [3] at the level of 95% in Fig. 9 to demonstrate the statistical significance of the results.

On the cyber-security problem, we observe an unexpectedly good performance of the surrogate neural network model with the adaptive auxiliary set. In all but *very hard* problem difficulty, the proposed method beats all other tested anomaly detectors including the source 5NN. 5NN remains notably best of all in the *very hard* problem difficulty. Here, the surrogate model still is the second best. Overall the surrogate model succeeds in retaining the average AUC of 0.941, which equals a drop of only 0.004 when compared to the average AUC 0.945 of 5NN detector (the average was computed over all problem difficulties).

The success of the surrogate model with the adaptive auxiliary set is significant (cf. Fig. 9). The reasoning about why a surrogate model can surpass the accuracy of a source AD may relate to the more general question of why and when parametric models can surpass nonparametric ones. Our results on *easy* and *medium* problems appear consistent with the known observation that parametric models tend to generalize better, especially on simpler problems (cf., e.g., [44]).

In the other cases, we observed, as expected, an accuracy slightly below or on par with the source anomaly detectors. To illustrate this we have performed one additional experiment. We constructed a surrogate neural network with IF as the source detector and compared the two. The results are included in Table 1. In this case, the AUC of the surrogate model is on average 0.014 lower than the AUC of IF, with no observed improvement in any of the problem difficulties.

4.4 Robustness of surrogate detectors

Surrogate neural network-based detectors depend on a number of parameters. The neural model itself depends on all the standard parametrization common in neural networks, the analysis of which is not the subject of this paper (see Sect. 4.2.2 for details of how parameters are set in this paper).

The surrogate neural detector additionally depends on the properties of the auxiliary data set used in its training. In the following, we discuss their impact.

4.4.1 Auxiliary set size

An important question concerns the required size of the auxiliary data set.

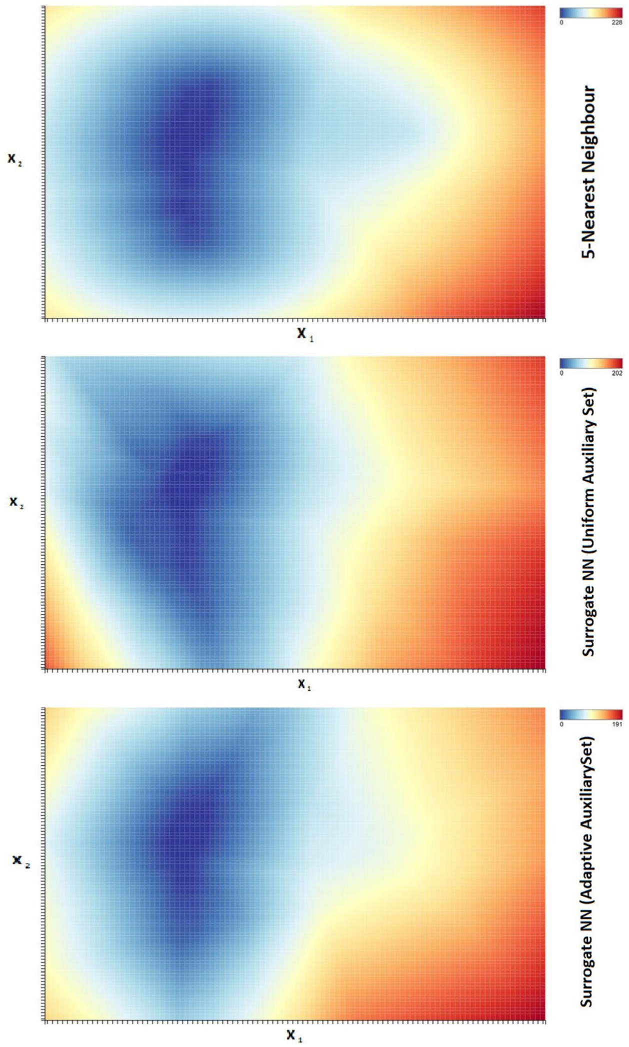

Fig. 8 Heat map illustration of anomaly scores induced on computer network security data set by (top to bottom): 5NN anomaly detector, surrogate neural network (with 5NN source detector) using the uniform auxiliary set, surrogate neural network (with 5NN source detector) using the adaptive auxiliary set. 2D projection to the first two principal components. Warmer color depicts a higher anomaly

Clearly, the auxiliary set needs to be sufficiently large to replicate the space of anomaly scores induced by the source detectors. We cannot give a universal answer due to the variety of scenarios where surrogate neural network detectors can be applicable. Instead, we show the dependence of surrogate detector accuracy on the auxiliary set size in the case of our cyber-security problem. In Fig. 10, the x-axis shows the ratio of the auxiliary set size to the input training data size and the y-axis shows the achieved AUC. It can be seen that already a surprisingly small number of auxiliary samples (equal to 5% of the input data set size) proved sufficient to enable very good eventual accuracy on the network security data set (cf. Sect. 4.1).

Table 1 Comparison of best achieved accuracy (AUC of ROC, scaled to [0,100], averaged over 8 runs) by each anomaly detector on network security data set

	Easy	Med	Hard	VHard	*Avg*
5NN	96.0	94.9	96.2	**90.8**	***94.5***
Surrogate-5NN-U	94.0	90.3	79.4	65.9	*82.4*
Surrogate-5NN-A	**98.7**	**97.9**	**96.7**	83.2	*94.1*
AE	94.5	92.9	92.2	80.3	*90.0*
LOF	97.1	92.8	90.1	75.2	*88.8*
IF	95.9	94.1	90.4	79.3	*89.9*
Surrogate-IF-A	94.4	92.4	89.9	77.4	*88.5*
GAN (Zenati)	87.0	85.2	87.4	71.7	*82.8*
GANomaly	96.1	93.7	93.7	73.2	*89.2*
Parzen	95.3	94.8	94.1	79.9	*91.0*

Results grouped by problem difficulty. Suffix U marks surrogate model with uniform auxiliary set, A marks adaptive auxiliary set

Best overall result per difficulty is emphasised in bold

To complement the picture, we include Fig. 11 to show the growth of surrogate model accuracy depending on the growing auxiliary set size on the benchmark Abalone data set (cf. Sect. 1). Even in this case, it can be seen that only a fractional auxiliary set size when compared to the size of the input data is sufficient to achieve very good accuracy. Figure 11 also illustrates the efficiency of adaptive auxiliary sets (cf. Sect. 3.1.2) in contrast to uniform auxiliary sets (cf. Sect. 3.1.1).

4.4.2 Adaptive auxiliary set parametrization

The adaptive auxiliary set construction procedure uses parameter k_{var} (variance multiplicative coefficient, see Sect. 3.1.2) which is essential to achieve optimal surrogate anomaly detector accuracy. The variance estimated from input data for the purpose of Parzen window sizing does not necessarily lead to the best possible auxiliary set. Therefore, for the auxiliary data set generation purpose, we multiply the estimated variance by the k_{var} coefficient, which needs to be optimized for each problem separately. The impact of various coefficient values is illustrated in Fig. 12 on the network security problem. Note that different levels of problem difficulty (cf. Sect. 4.1.1) may require different k_{var} values. In the experimental evaluation (cf. Sect. 4.3), however, we fixed one parameter only for all levels of difficulty.

4.4.3 Adaptive auxiliary set efficiency

The auxiliary set construction procedure as described in Sect. 3.1.2 has been experimentally shown to provide results competitive with benchmark anomaly detectors (see

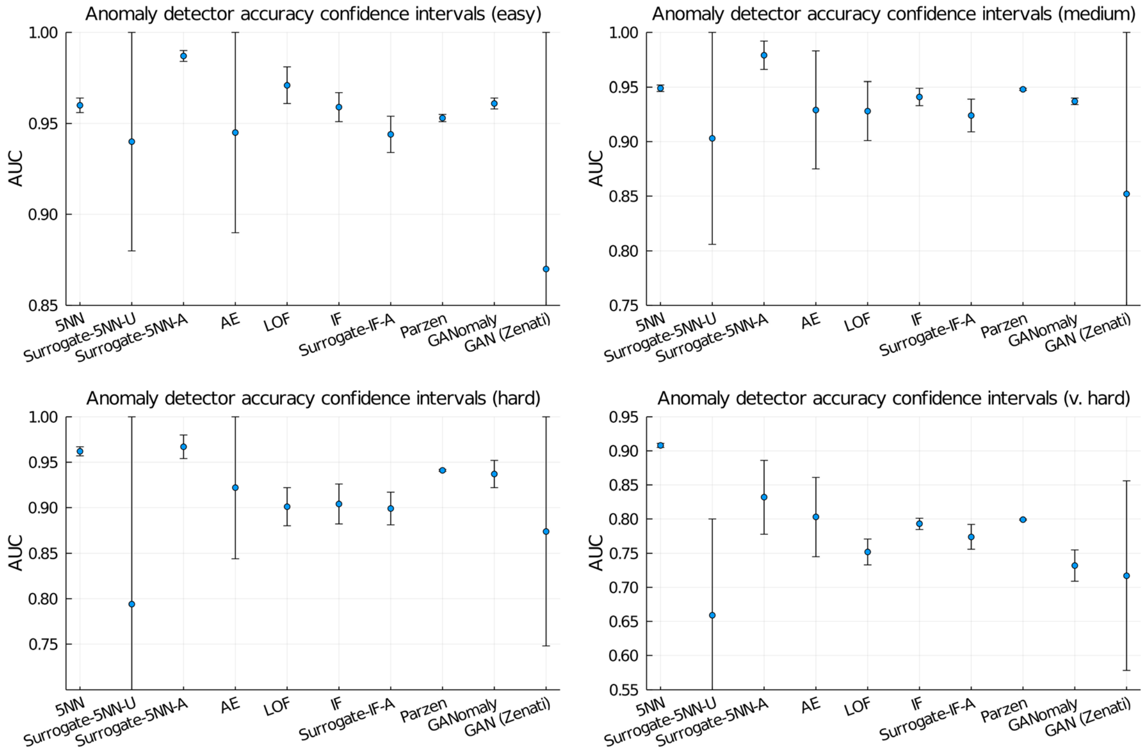

Fig. 9 Comparing the accuracy achieved by the baseline and surrogate anomaly detectors on the network security problem. Confidence intervals for mean AUC of ROC at 95% level. Four problem difficulties (top left to bottom right): *easy*, *medium*, *hard*, *very hard*. Compare particularly the 5NN anomaly detector to the respective surrogate neural network with adaptive auxiliary set *Surrogate-5NN-A*

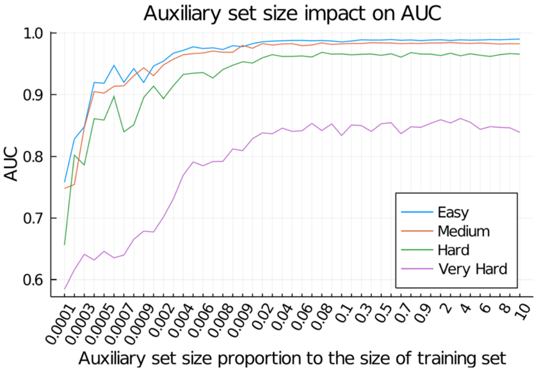

Fig. 10 Accuracy of surrogate neural network detector depending on the adaptive auxiliary set size. Network security data set. Note that very small auxiliary set (about 5% of the input data size) can suffice to achieve best effect

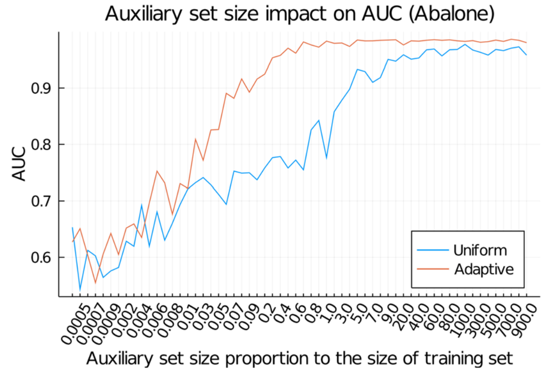

Fig. 11 Accuracy of surrogate neural network detector depending on the auxiliary set size. Abalone data set, accuracy averaged over all levels of difficulty. Note the higher efficiency of the *adaptive* over the *uniform* auxiliary set

Sect. 4.3.2). Let us discuss the question of whether there is still space for its improvement.

Arguably, for the purpose of AD, the most important regions in the modeled space may be those with a lower density of input samples. It appears that such regions would benefit from a higher density of generated auxiliary samples than regions where the mass of input samples lies. The adaptive auxiliary set generating procedure, however, tends to produce the opposite. It generates more auxiliary samples and thus captures more details of the input distribution for areas of high input data density while areas of low density and singular samples get covered by fewer auxiliary samples.

Based on this observation, we implemented two modifications of the generation algorithm: 1. the relative number of auxiliary samples generated per Parzen window is made inversely proportional to the baseline anomaly score of the Parzen window center point (this

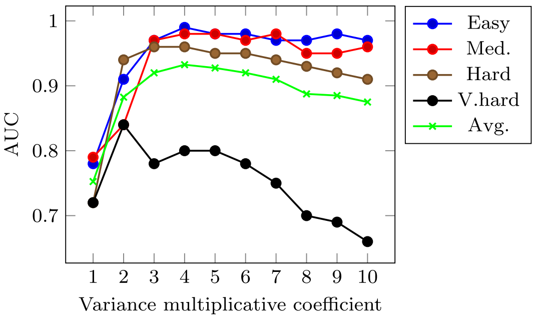

Fig. 12 Surrogate neural network detector accuracy (AUC of ROC) depends on a parameter *variance multiplicative coefficient* k_{var} if the adaptive auxiliary set is used

effect is controlled through multiplication by constant σ where $\sigma = 1$ is equivalent to the original algorithm), 2. we also made the Parzen window width inversely proportional to the baseline anomaly score (this effect is controlled through multiplication by constant δ where $\delta = 1$ is equivalent to the original algorithm).

We performed a large number of tests for a grid of values $\sigma \in \{1, \ldots, 50\}$ and $\delta \in \{0.1, \ldots, 10\}$. In Fig. 13, we show the effect on four illustrative examples (compare to the baseline adaptive method result in Fig. 2). We do not include more details on the results because in all cases the resulting surrogate neural network detector accuracy dropped below the baseline adaptive method defined in Sect. 3.1.2.

We also tested the impact on auxiliary data set size efficiency (compare to Sect. 4.4.1). Again, no improvement has been reached. Therefore, the default adaptive auxiliary set generation procedure (cf. Sect. 3.1.2) remains the recommended option.

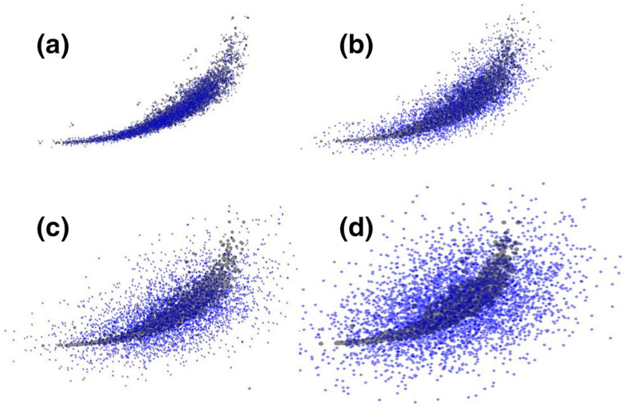

Fig. 13 Modified adaptive auxiliary set generation procedure redistributes auxiliary samples from more dense to less dense regions of the input space. Images illustrate modified auxiliary set variants from a) those with reduced window size in anomalous regions $\delta \ll 1$ to d) those with enlarged window size and weight in anomalous regions $\sigma > 1$ and $\delta \gg 1$

5 Discussion

The idea of constructing a surrogate neural network detector to replicate a source anomaly detector (which uses a non-neural model) has been motivated by the intention to improve inference speed in a large-scale industrial setting. We have shown the usefulness of the idea on a real cyber-security problem.

We have observed that the effect of introducing a surrogate neural network can have a positive impact also on accuracy, although such an effect cannot be guaranteed. We have also observed that the size of the adaptive auxiliary set can be considerably smaller than the size of the input data set, without negative impact. In general, it should be expected though that surrogate models achieve comparable or slightly worse accuracy than the source anomaly detector.

The surprising result that a shallow surrogate neural network could over-perform deep neural models (see Table 1) is a direct consequence of the fact that its source AD the *k*NN performed better than deep models on our cyber-security problem. Arguably, the high expressivity of deep models can be a disadvantage in a setting where generalization is very difficult (see particularly VHard in Table 1).

Note that for other problem areas the idea of surrogate anomaly detectors can be decoupled to two separate parts: construction of auxiliary training set and training a model on top of it.

Once an auxiliary training set is constructed from a source anomaly detector, it is imaginable to train a non-neural model on top of it. We have not investigated such an option further.

Another interesting option is to utilize auxiliary sets for collecting information from multiple source anomaly detectors. We discuss this option in more detail in the following.

5.1 Fusion of multiple anomaly detectors

Many different types of anomaly detector ensembles have been proposed in the literature [14, 42, 54, 50, 64] to mitigate limitations of individual detectors. Specifically in the area of our practical interest—in network security—ensembles of predictors are commonly applied [57].

The flexibility of the auxiliary set construction procedure trivially enables fusing outputs from multiple baseline detectors into a single auxiliary set. We envisage multiple implications as follows.

5.1.1 Mitigating the problem of incorrect parametrization

Fusing outputs from different instances of the same type of detector can help smooth out the impact of potentially incorrect parametrization. This may become useful if there is uncertainty about which parameters to choose. In Tab. 2, we illustrate this effect by fusing a number of kNN detectors for multiple different values of k. Fusion in this case does not lead to the overall best accuracy but provides better accuracy than is the average of individual accuracies of the fused detectors (note the last two pairs of rows in the table).

5.1.2 Fusing detectors to reduce ensemble complexity

A practical use of detector fusion can expectably be the option to train a single surrogate neural network detector to replace an ensemble of detectors. Especially in the case of large ensembles or ensembles of detectors of mixed types, the advantage can be not just the expected inference speedup, but also the simplification of the overall deployed anomaly detection system.

The problem is that various source detectors may provide anomaly scores at different intervals or even unbounded. The prerequisite to their fusion therefore would be the normalization of the individual detectors' output. Normalization is possible in multiple ways. Platt scaling [43] can be considered. As a simpler option (inspired by the discussion in [41]) we propose the following.

Assuming we have a training set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ where $\mathbf{x}_i \in \mathbb{R}^d$, $\forall i \in \{1, \ldots, n\}$ and the corresponding anomaly scores obtained from a generic anomaly detector $Y = \{y_1, y_2, \ldots, y_n\}$ where $y_i \in \mathbb{R}$, $\forall i \in \{1, \ldots, n\}$, the normalized anomaly score vector $\bar{Y}$ is obtained as

$$\bar{y_i} = \frac{L(y_i)}{n}, \quad \forall i \in \{1, \ldots, n\}$$

where

$$L(y) = \sum_{i=1}^{n} \begin{cases} 1, & \text{if } y_i < y \\ 0, & \text{otherwise} \end{cases}$$

Assuming a finite size of $\mathbf{X}$, the normalized anomaly score for sample $\mathbf{x}_i$ equals to the proportion of samples in $\mathbf{X}$ with lower anomaly scores than is the anomaly score of $\mathbf{x}_i$. The normalized scores are then from [0, 1).

Table 2 Accuracy of fused anomaly detectors compared to individual detectors (AUC of ROC, scaled to [0,100])

Detector	Easy	Med	Hard	VHard
1NN	68.29	75.79	66.55	62.57
3NN	86.48	92.25	90.63	72.78
5NN	98.72	97.89	96.66	83.24
7NN	98.44	98.16	95.96	83.62
9NN	98.63	98.22	95.65	83.57
avg 3NN,5NN,7NN	94.55	96.10	**94.42**	79.88
fused (3,5,7)-NN	**98.07**	**97.21**	94.38	**81.23**
avg 1NN,3NN...9NN	90.11	92.46	89.09	77.16
fused (1,3,5,7,9)NN	**97.86**	**94.39**	**94.08**	**79.47**

Note that fusing various kNN detectors into a single surrogate anomaly detector can lead to better accuracy than is the average accuracy over the various standalone detectors. When comparing averaged versus the respective fused accuracy, the better result is emphasised in bold

5.1.3 Fusing detectors to optimize response

The most complex fusion that we envisage should enable optimization of anomaly detection accuracy locally across the input sample space. It is based on the observation that principally different detection models are likely to have different strengths and weaknesses, presumably in different parts of the input space. Let us assume that for each detector in a collection of various detectors it is possible to estimate the confidence of its output for a specific sample. Then, using the normalization (see Sect. 5.1.2) an auxiliary set can be constructed from outputs of all the detectors, where the contribution of each detector to a single auxiliary sample is conditioned by the detector's sufficient confidence. In this way, various detectors from the collection would cover various parts of the auxiliary space, presumably leading to a more robust surrogate anomaly detector. The prerequisite here would be the ability to evaluate the confidence of each considered source detector. We refer to [41] for a solution to this problem.

6 Conclusion

Motivated by the needs of large-scale cyber-security systems we addressed the problem of anomaly detection inference speed. We proposed to construct surrogate neural network anomaly detectors to replace existing slow anomaly detectors or detector ensembles. We have shown that simple neural network formalism can be used to solve this problem. We have shown that it is possible to construct fast surrogate anomaly detectors without notable loss of accuracy. We have shown that the idea of surrogate anomaly detectors can also enable simplification of deployed anomaly detection systems, especially in the case of ensembles. We have observed that at least in network security the use of surrogate neural network detectors can occasionally improve the accuracy of the best baseline anomaly detectors.

Funding This work has been supported by the Grant Agency of the Czech Technical University in Prague, Grant No. SGS20/188/OHK4/3T/14.

Declarations

Conflicts of interest The authors declare no conflicts of interest. All authors have seen the manuscript and approved the submission to the journal. We confirm that the content of the manuscript has not been published or submitted for publication elsewhere.

References

1. Akcay S, Atapour-Abarghouei A, Breckon TP (2019) Ganomaly: semi-supervised anomaly detection via adversarial training. In: Jawahar CV, Li H, Mori G, Schindler K (eds) Computer vision—ACCV 2018. Springer, Cham, pp 622–637
2. Aleskerov E, Freisleben B, Rao B (1997) Cardwatch: a neural network based database mining system for credit card fraud detection. In: Proceedings of the IEEE/IAFE 1997 computational intelligence for financial engineering, pp 220–226. https://doi.org/10.1109/CIFER.1997.618940
3. Altman D, Machin D, Bryant T, Gardner M (2013) Statistics with confidence: confidence intervals and statistical guidelines. Wiley
4. An J, Cho S (2015) Variational autoencoder based anomaly detection using reconstruction probability. Technical report
5. Angiulli F, Pizzuti C (2002) Fast outlier detection in high dimensional spaces. In: European conference on principles of data mining and knowledge discovery, pp 15–27. Springer
6. Bentley JL (1975) Multidimensional binary search trees used for associative searching. Commun ACM 18(9):509–517
7. Bergman L, Cohen N, Hoshen Y (2020) Deep nearest neighbor anomaly detection. arXiv preprint arXiv:2002.10445
8. Beygelzimer A, Kakade S, Langford J (2006) Cover trees for nearest neighbor. In: Proceedings of the 23rd international conference on Machine learning, pp 97–104. ACM
9. Breunig MM, Kriegel HP, Ng RT, Sander J (2000) Lof: identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD international conference on Management of data, pp 93–104
10. Brown CD, Davis HT (2006) Receiver operating characteristics curves and related decision measures: a tutorial. Chem. Intel. Lab. Syst. 80(1):24–38
11. Cannady J (1998) Artificial neural networks for misuse detection. In: National information systems security conference, pp 368–81
12. Chalapathy R, Chawla S (2019) Deep learning for anomaly detection: a survey
13. Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: a survey. ACM Comput Surv (CSUR) 41(3):15
14. Chiang A, Yeh YR (2015) Anomaly detection ensembles: In defense of the average. In: 2015 IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology (WI-IAT), vol 3, pp 207–210. IEEE
15. Dau HA, Ciesielski V, Song A (2014) Anomaly detection using replicator neural networks trained on examples of one class. In: Asia-Pacific conference on simulated evolution and learning, pp 311–322. Springer
16. Demuth HB, Beale MH, De Jess O, Hagan MT (2014) Neural network design. Martin Hagan
17. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7, 1–30. http://dl.acm.org/citation.cfm?id=1248547.1248548
18. Dua D, Graff C (2017) UCI machine learning repository. http://archive.ics.uci.edu/ml
19. Emmott AF, Das S, Dietterich T, Fern A, Wong WK (2013) Systematic construction of anomaly detection benchmarks from real data. In: Proceedings of the ACM SIGKDD workshop on outlier detection and description, ODD ’13, pp 16–21. ACM, New York, NY, USA. https://doi.org/10.1145/2500853.2500858
20. Flusser M, Pevný T, Somol P (2018) Density-approximating neural network models for anomaly detection. In: ACM SIGKDD workshop on outlier detection de-constructed. London, United Kingdom
21. Flusser M, Somol P (2021) Adaptive approach for density-approximating neural network models for anomaly detection. In: Herrero Á, Cambra C, Urda D, Sedano J, Quintián H, Corchado E (eds) 13th international conference on computational intelligence in security for information systems (CISIS 2020). Springer, Cham, pp 415–425
22. Friedman JH, Bentley JL, Finkel RA (1977) An algorithm for finding best matches in logarithmic expected time. ACM Trans Math Softw (TOMS) 3(3):209–226
23. Garcia S, Derrac J, Cano J, Herrera F (2012) Prototype selection for nearest neighbor classification: taxonomy and empirical study. IEEE Trans Pattern Anal Mach Intel 34(3):417–435. https://doi.org/10.1109/TPAMI.2011.142
24. Goodfellow I, Bengio Y, Courville A (2016) Deep larning. MIT Press. http://www.deeplearningbook.org
25. Goyal S, Raghunathan A, Jain M, Simhadri HV, Jain P (2020) Drocc: deep robust one-class classification. In: International conference on machine learning, pp 3711–3721. PMLR
26. Grill M, Pevnỳ T (2016) Learning combination of anomaly detectors for security domain. Comput Networks 107:55–63
27. Grim J, Somol P, Haindl M, Danes J (2009) Computer-aided evaluation of screening mammograms based on local texture models. IEEE Trans Image Process 18(4):765–773. https://doi.org/10.1109/TIP.2008.2011168
28. Gu X, Akoglu L, Rinaldo A (2019) Statistical analysis of nearest neighbor methods for anomaly detection. arXiv preprint arXiv:1907.03813
29. Hariri S, Carrasco Kind M, Brunner RJ (2019) Extended isolation forest. IEEE Trans Knowl Data Eng, p 1–1. https://doi.org/10.1109/tkde.2019.2947676
30. Hendrycks D, Mazeika M, Dietterich T (2018) Deep anomaly detection with outlier exposure. arXiv preprint arXiv:1812.04606
31. Jiang W, Hong Y, Zhou B, He X, Cheng C (2019) A gan-based anomaly detection approach for imbalanced industrial time series. IEEE Access 7:143608–143619. https://doi.org/10.1109/ACCESS.2019.2944689
32. Kim J, Scott CD (2012) Robust kernel density estimation. J Mach Learn Res 13(Sep), 2529–2565
33. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980
34. Kohout J, et al. (2016) Detection of malicious network connections. https://patents.google.com/patent/US9344441B2/. Cisco Technology, Inc., San Jose, CA (US), US Patent 9,344,441 B2
35. Kriegel HP, Kröger P, Schubert E, Zimek A (2009) Loop: local outlier probabilities. In: Proceedings of the 18th ACM conference on information and knowledge management, pp 1649–1652
36. Liu FT, Ting KM, Zhou ZH (2008) Isolation forest. In: 2008 eighth IEEE international conference on data mining, pp 413–422. IEEE
37. Loader CR (1996) Local likelihood density estimation. Ann Statist 24(4):1602–1618. https://doi.org/10.1214/aos/1032298287
38. Mika S, Schölkopf B, Smola AJ, Müller KR, Scholz M, Rätsch G (1999) Kernel PCA and de-noising in feature spaces. In: Advances in neural information processing systems, pp 536–542

39. Mittal S (2019) A survey on optimized implementation of deep learning models on the nvidia jetson platform. J Syst Arch 97:428–442. https://doi.org/10.1016/j.sysarc.2019.01.011. https://www.sciencedirect.com/science/article/pii/S1383762118306404
40. Mukkamala S, Janoski G, Sung A (2002) Intrusion detection using neural networks and support vector machines. In: Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on, vol 2, pp 1702–1707. IEEE
41. Perini L, Vercruyssen V, Davis J (2020) Quantifying the confidence of anomaly detectors in their example-wise predictions. In: The European conference on machine learning and principles and practice of knowledge discovery in databases. Springer
42. Pevný T (2016) Loda: lightweight on-line detector of anomalies. Mach Learn 102(2):275–304
43. Platt J et al (1999) Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Adv Large Margin Class 10(3):61–74
44. Russel SJ, Norvig P (2014) Artificial intelligence: a modern approach. Pearson Education Limited, UK
45. Ryan J, Lin MJ, Miikkulainen R (1998) Intrusion detection with neural networks. In: Advances in neural information processing systems, pp 943–949
46. Sakurada M, Yairi T (2014) Anomaly detection using autoencoders with nonlinear dimensionality reduction. In: Proceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis, MLSDA'14, pp 4:4–4:11. ACM, NY, USA. https://doi.org/10.1145/2689746.2689747
47. Sarasamma ST, Zhu QA, Huff J (2005) Hierarchical kohonenen net for anomaly detection in network security. IEEE Tran Syst Man Cybern Part B 35(2):302–312
48. Schlegl T, Seeböck P, Waldstein SM, Langs G, Schmidt-Erfurth U (2019) f-anogan: fast unsupervised anomaly detection with generative adversarial networks. Med Image Anal 54:30–44. https://doi.org/10.1016/j.media.2019.01.010
49. Schölkopf B, Platt JC, Shawe-Taylor J, Smola AJ, Williamson RC (2001) Estimating the support of a high-dimensional distribution. Neural comput 13(7):1443–1471
50. Shoemaker L, Hall LO (2011) Anomaly detection using ensembles. In: International workshop on multiple classifier systems, pp 6–15. Springer
51. Škvára V, Franců J, Zorek M, Pevný T, Šmídl V (2021) Comparison of anomaly detectors: context matters. IEEE Trans Neural Networks Learn Syst 33(6):2494–2507. https://doi.org/10.1109/TNNLS.2021.3116269
52. Škvára V, Pevný T, Šmídl V (2018) Are generative deep models for novelty detection truly better?
53. Staerman G, Mozharovskyi P, Clémençon S, d'Alché Buc F (2019) Functional isolation forest
54. Tama BA, Nkenyereye L, Islam SR, Kwak KS (2020) An enhanced anomaly detection in web traffic using a stack of classifier ensemble. IEEE Access 8:24120–24134
55. Ting KM, Zhu Y, Zhou ZH (2018) Isolation kernel and its effect on svm. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining, pp 2329–2337
56. Uhlmann JK (1991) Satisfying general proximity/similarity queries with metric trees. Inf Process Lett 40(4):175–179
57. Vanerio J, Casas P (2017) Ensemble-learning approaches for network security and anomaly detection. In: Proceedings of the workshop on big data analytics and Machine learning for data communication networks, pp 1–6
58. Vincent P, Larochelle H, Bengio Y, Manzagol PA (2008) Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th international conference on machine learning, pp 1096–1103. ACM
59. Vincent P, Larochelle H, Lajoie I, Bengio Y, Manzagol PA (2010) Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. J Mach Learn Res 11(Dec):3371–3408
60. Yeung DY, Chow C (2002) Parzen-window network intrusion detectors. In: Object recognition supported by user interaction for service robots, vol 4, pp 385–388. IEEE
61. Zenati H, Foo CS, Lecouat B, Manek G, Chandrasekhar VR (2018) Efficient gan-based anomaly detection. CoRR abs/1802.06222. arXiv:1802.06222
62. Zhai S, Cheng Y, Lu W, Zhang Z (2016) Deep structured energy based models for anomaly detection. In: Proceedings of the 33rd international conference on international conference on machine learning, Vol 48, ICML'16, pp 1100–1109. JMLR.org. http://dl.acm.org/citation.cfm?id=3045390.3045507
63. Zhao M, Saligrama V (2009) Anomaly detection with score functions based on nearest neighbor graphs. In: Advances in neural information processing systems, pp 2250–2258
64. Zhao Z, Mehrotra KG, Mohan CK (2015) Ensemble algorithms for unsupervised anomaly detection. In: International conference on industrial, engineering and other applications of applied intelligent Systems, pp 514–525. Springer