An Encoding Adversarial Network for Anomaly Detection

Elies gherbi ELIESGHERBI@GMAIL.COM ELIES.GHERBI@IRT-SYSTEMX.FR
Irt Systemx, 91120 palaiseau, France

Blaise Hanczar BLAISE.HANCZAR@UNIV-EVRY.FR
IBISC, Univ Evry, Université Paris-Saclay

Jean-Christophe Janodet JEANCHRISTOPHE.JANODET@UNIV-EVRY.FR
IBISC, Univ Evry, Université Paris-Saclay

Witold Klaudel WITOLD.KLAUDEL@IRT-SYSTEMX.FR
Irt Systemx, 91120 palaiseau, France

Editors: Wee Sun Lee and Taiji Suzuki

Abstract

Anomaly detection is a standard problem in Machine Learning with various applications such as health-care, predictive maintenance, and cyber-security. In such applications, the data is unbalanced: the rate of regular examples is much higher than the anomalous examples. The emergence of the Generative Adversarial Networks (GANs) has recently brought new algorithms for anomaly detection. Most of them use the generator as a proxy for the reconstruction loss. The idea is that the generator cannot reconstruct an anomaly. We develop an alternative approach for anomaly detection, based on an Encoding Adversarial Network (AnoEAN), which maps the data to a latent space (decision space), where the detection of anomalies is done directly by calculating a score. Our encoder is learned by adversarial learning, using two loss functions, the first constraining the encoder to project regular data into a Gaussian distribution and the second, to project anomalous data outside this distribution. We conduct a series of experiments on several standard bases and show that our approach outperforms the state of the art when using 10% anomalies during the learning stage, and detects unseen anomalies.

Keywords: Anomaly Detection, Deep Learning, Encoding Network, Adversarial Learning, Generative Adversarial Networks, Intrusion detection, week supervised learning, Unbalanced datasets.

1. Introduction

Anomaly detection is a well-established topic in Machine Learning, with many applications in domains such as fraud detection, cybersecurity, video surveillance, and predictive maintenance (Chandola et al., 2009; Kiran et al., 2018; Chalapathy and Chawla, 2019). Moreover, the recent advances in Artificial Intelligence bring insight into future applications, as autonomous transportation like self-driving cars. Many cyber-security threats can impact the usability of those systems. The problem of anomaly detection highlights many risks related to those threats and can help the standard security systems to face new threats.

We can formulate the anomaly detection problem as follows. Let D be a data set containing a large number of normal examples (the normal states of the system) X_n, and a relatively small number of anomalous examples X_a. A model M must learn the distribution function p_X over the normal data during training. Then, given any test example x, it must determine whether x deviates from the learned distribution p_X by using an anomaly score function $a(x)$.

Following Chalapathy and Chawla (2019), there exist three settings for anomaly detection. The first is the unsupervised case, where an algorithm has to discover intrinsic properties of the data to detect the anomalies without any label guidance(Campello et al., 2015); The training set contains both normal and anomalous examples, but the labels are not available. The second is the supervised case, where an algorithm must usually face unbalanced datasets, with significant rates of normal data and few anomalous examples. The third is the one-class classification, where the algorithm has access only to a large set of normal data. Unlike the unsupervised case, there are no anomalous examples in the training set. In this paper, we focus on both the supervised and one-class anomaly detection problems. Notice that even though few anomalous examples may exist in the data, supervised learning algorithms are still challenging, because the set of anomalies often does not form a homogeneous class, that is, a set of examples that shares some semantics.

A large number of papers propose machine learning-based anomaly detection models (Chandola et al., 2009; Hodge and Austin, 2004; Pimentel et al., 2014). The recent advances in the field of deep learning have made it possible to revise this problem (Chalapathy and Chawla, 2019; Kiran et al., 2018) and in many application domains like intrusion detection system Galinina et al. (2018). In particular, Generative Adversarial Networks (GANs) (Goodfellow et al., 2014a,b), that were proved very efficient in many application fields (Creswell et al., 2018), have also been adopted in recent works on anomaly detection (Schlegl et al., 2017; Zenati et al., 2018; Akcay et al., 2018; Golan and El-Yaniv, 2018; Sabokrou et al.; Deecke et al.).

In this article, we investigate several problems related to GAN-based anomaly detection methods. We propose a new method, called AnoEAN (Encoding Adversarial Network for Anomaly Detection). The principle of AnoEAN is to learn a function (Encoder) that projects the original dataset into a small dimension latent space so that the normal examples are projected in a restricted region of the latent space and the anomalies outside this region. This latent representation allows us to identify anomalies directly in the latent space by using a Mahalanobis distance on the distribution induced by the normal examples. We thus eliminate all the problems related to the reconstruction loss function. To do so, we develop a new approach that trains an encoder by adversarial learning. We assume that we have a large amount of normal data and a small number of anomalies, which is a common framework for anomaly detection. We finally conduct a series of experiments proving that AnoEAN performs better than conventional anomaly detection techniques, including those based on GANs, using both the MNIST base of handwritten digits (LeCun, 1998) and two standard network intrusion detection databases (KDD'99, NSL-KDD).

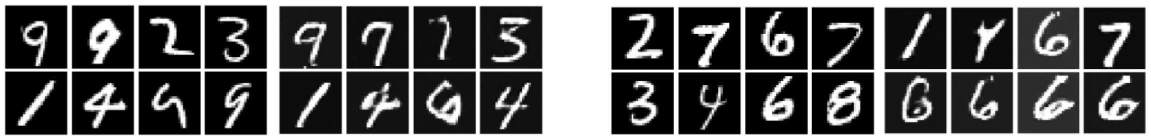
2. Related works

According to the taxonomy of Golan and El-Yaniv (2018), there are two leading families of methods. The first category, called Representation learning, consists of learning a representation of the normal data, which is used to trigger an alarm when deviant behavior is detected. Many machine learning techniques have been used to construct such models, including one-class SVM (OCSVM) (Schölkopf et al., 2000). The problem returns to find a hypersphere with a small radius that encloses the majority of the normal data and keeps anomalies outside. Contrary to the deep learning approaches, OCSVM and other classic machine learning techniques require a specific feature preprocessing. The kernel methods also suffer from quadratically scaling, and require the storage of the support vectors which can be restrictive for an embedded system. Notice that Ruff et al. (2018); Chalapathy et al. (2018) has proposed a deep learning adaptation of one-class classification that does not suffer from these limitations.

TThe second category is based on a Reconstruction loss anomaly score. The model constructs a compact representation of the characteristics of the normal examples and learns to reconstruct the examples from this representation. These methods are based on the assumption that it is more difficult to reconstruct an anomaly than normal examples. The autoencoders were the first to be developed in this framework (Xia et al., 2015; Xu et al., 2018), but the adversarial learning techniques became more and more popular, and several GAN variants have been adapted to tackle anomaly detection problems (Schlegl et al., 2017; Zenati et al., 2018; Zenati et al.; Akcay et al., 2018; Golan and El-Yaniv, 2018; Sabokrou et al.; Deecke et al.).

A Generative Adversarial Network (GAN) is composed of two neural networks: a generator G, that transforms a vector z drawn from simple prior distribution (latent space) into an artificial data space (same dimension as a real data space), and a discriminator D whose role is to differentiate artificial data from real data. One drives such a system competitively: the generator must deceive the discriminator by implicitly learning to approximate the distribution of real data (Goodfellow et al., 2014a). In the context of anomaly detection, AnoGAN (Schlegl et al., 2017) and its variants (Deecke et al.), learn a GAN from the normal data. Then, for a given test example x, the algorithm looks for a point z in the latent space such that $G(z) \approx x$; if it fails, then the example x is considered as an anomaly. In other words, the GAN-based approaches work upon the inversion of the generator: AnoGAN tries to compute $z = G^{-1}(x)$ under the assumption that G is invertible only for normal examples. Nevertheless, AnoGAN does not explicitly invert the generator: a reconstruction loss function is defined, typically $\mathcal{L}_r = \|x - G(z)\|$, and the techniques consist in finding the point z which minimizes $\mathcal{L}_r$ withe respect to z. The process is very time-consuming because a gradient descent is performed for each test example. Notice that Creswell and Bharath (2018) improved this idea by inverting many images at once. The GANs also suffer from the mode collapse (Salimans et al., 2016): The generator may have a limited diversity during the inference phase regardless of the input. In practice, it means that there exists a single point that the generator thinks as the most optimal point to generate, whatever the input. See Fig. 1c, 1d.

In order to get around these problems, other algorithms like EGBAD and ALAD (Zenati et al., 2018; Zenati et al.) learn Bidirectional GANs (BiGANs) (Donahue et al., 2016; Dumoulin et al., 2016; Li et al., 2017) rather than GANs. Their architectures are composed of three networks: a generator G and a discriminator D as before, and a third network called an encoder E, whose role is to project the data to the latent space with $E = G^{-1}$. The encoder is learned alongside the GAN. Moreover, in this setting, finding a point z such that $G(z) \approx x$ is immediate since we have $z = E(x)$. However, training an additional encoder network may encourage over-fitting, resulting in poor reconstruction (Creswell and Bharath,



(*a*) original data (*b*) reconstructed data (*c*) original data (*d*) reconstructed data

Figure 1: (a) and (b) shows the non-identifiability issue: the generator performs well in reconstruction, but the encoder has difficulties to retrieve the correct pairs between the data space and latent space. (c) and (d) shows the mode collapse issue: the generator is more likely to project every latent point to digit (6), which proves that the generator lacks from diversity.

2018). Moreover, in practice, the generator and the encoder are rarely exact inverses of each other: they are inverse from a theoretical standpoint, after convergence. Furthermore, the BiGANs may suffer from non-identifiability issues, as shown by Li et al. (2017): a single instance z can potentially correspond to many possible values of x (see Fig. 1a, 1b). To deal with this problem, ALICE (Li et al., 2017) uses conditional entropies to control the uncertainty of a pair (z, x). More precisely, the authors bound the conditional entropy by using the criterion of cycle-consistency (Zhao et al.).

Most of the GAN-based approaches work upon the assumption that the generator is invertible. The inverse of the generator is obtained either by driving another network which simulates the inverse of the generator or via an optimization process. However, it has been shown experimentally that in both cases, the approximated function is not the inverse of the generator everywhere (see Fig. 1a and 1b). Thus, applying the generator again on the antecedent of a datum x generally fails to reconstruct x, even though datum x was a normal example. In practice, this phenomenon dramatically increases the rate of false alarms.

3. AnoEAN algorithm

3.1. Formulation

We consider a problem in which we monitor the data $x \in \mathbb{R}^p$ describing the state of a system with p variables. Our goal is to trigger an alert when the data shows that the system is out of normal behavior. For this, we construct an anomaly score $a(x) : \mathbb{R}^p \to [0, +\infty[$ measuring the degree of anomaly of an example x, the normal example must have a score close to 0. To learn this function, we have a training set containing N examples corresponding to a normal behavior $\{x_{n_i} \mid i = 1..N\}$ and M examples corresponding to anomalies $\{x_{a_i} \mid i = 1..M\}$ with $M \ll N$. Notice that unlike normal data, the anomalies do not form a homogeneous class.

Our method, named AnoEAN for *Anormaly Detection by Encoding Adversarial Network*, consists in projecting the examples x_n and x_a into a small space, called decision space, in which the distribution of normal examples is Gaussian. We call *encoder*, the neural network that projects the examples from the original space into the decision space $E(x) : \mathbb{R}^p \to \mathbb{R}^d$ with $d \ll p$. Let p_z be a Gaussian distribution $\mathcal{N}(0, I)$ in the decision space. The purpose of the encoder is to project the normal examples in p_z and the anomalies outside p_z.

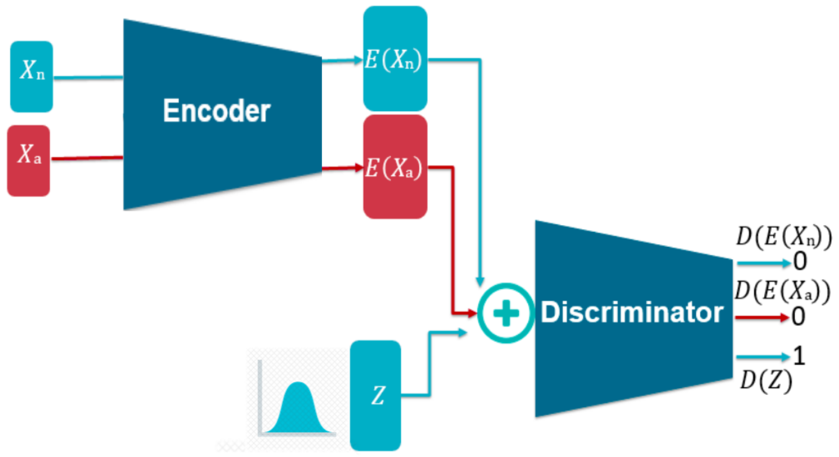


Figure 2: The training phase of AnoEAN: the Encoder and the Discriminator are adversarially learnt.

For this, in the same way as the GANs, we use a second network called *discriminator* $D(z) : \mathbb{R}^d \to \mathbb{R}$ which receives as input a vector of the decision space and predicts if this vector comes from p_z or from the encoder. $D(z)$ returns the probability that the vector z comes from p_z. The encoder must therefore both misleads the discriminator on the normal example projection $E(x_n)$ to make it believe that it comes from p_z and help the discriminator differentiate p_z from the projection of the anomalies $E(x_a)$. The architecture of our AnoEAN model is shown in Fig. 2. The discriminator and the encoder must respectively minimize the following L_D and L_E loss functions:

$$L_D = -\mathbf{E}_{z\sim p_z}[\log D(z)] - \mathbf{E}_{x_n\sim p_{x_n}}[\log(1 - D(E(x_n)))] - \mathbf{E}_{x_a\sim p_{x_a}}[\log(1 - D(E(x_a)))] \quad (1)$$

$$L_E = \mathbf{E}_{x_n\sim p_{x_n}}[\log(1 - D(E(x_n)))] + \mathbf{E}_{x_a\sim p_{x_a}}[\log(D(E(x_a)))] \quad (2)$$

In Eq. (1) and (2), the two first terms of loss function L_D, and the first term of loss function L_E are similar to the basic loss functions of a GAN. Instead of a noise vector in the input of the generator of the GAN, in AnoEAN a vector representing a normal example is in the input of the encoder. By maximizing $D(z)$ and minimizing $D(E(x_n))$, the decoder learns to differentiate vectors drawn from p_z and projections of normal examples $E(x_n)$. By maximizing $D(E(x_n))$ the encoder gets the distribution of the projections of normal examples $E(x_n)$ closer to p_z. If we do not add the terms $D(E(x_a))$ in L_D and L_E then there are no constraints on the projections of the anomalies, and the encoder tends to project both normal examples and anomalies on p_z. The consequence is that we can not differentiate normal examples from anomalies in the decision space. On the contrary, by minimizing $D(E(x_a))$, we get that (i) the decoder learns to differentiate vectors drawn from p_z and projections of normal examples $E(x_a)$, and (ii) the encoder also minimizes the overlap between the distribution of projections of the anomalies $E(x_a)$ and p_z.

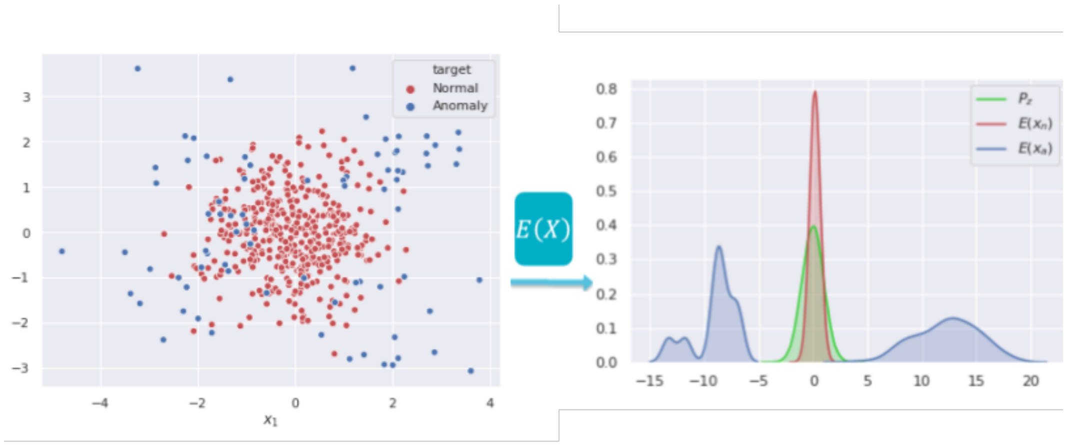


Figure 3: A sketch of the projection from the data space to the decision space that is performed by a well-trained Encoder.

Fig. 3 shows a toy dataset illustrating the objective of the encoder. The 2-dimension dataset (left panel) corresponds of 550 training examples; The normal examples are represented with 500 red points, which are drawn from a normal distribution; The remaining 50 blue points are drawn from multiple other Gaussian distributions and scattered around to simulate anomalous examples. The 1-dimension decision space is represented in the right panel, where the green, red and blue curves represent respectively the distribution of p_z, $E(x_n)$ and $E(x_a)$. The distribution of the $E(x_n)$ fits p_z and the overlap between the distribution of the $E(x_a)$ and p_z tends to 0. This clearly yields an efficient space in which the anomaly score can be computed.

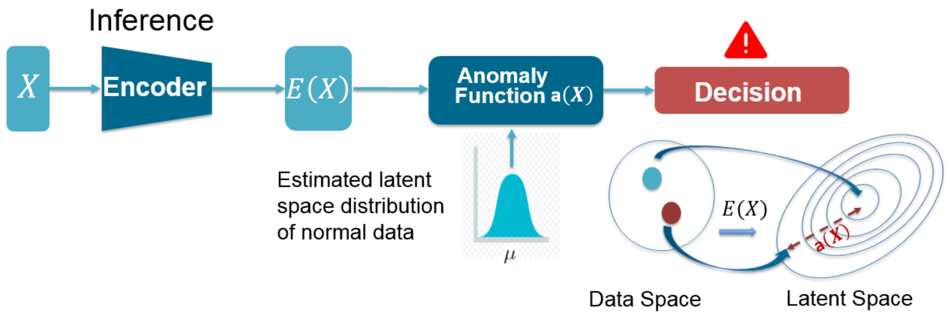


Figure 4: The inference (test) phase: only the Encoder is requested.

Once the model is trained, the prediction of the anomalies requires only the use of the encoder (Fig 4). The anomaly score for an example x is the distance between its projection in the decision space $E(x)$ and the distribution of the projection of the normal examples. This distribution is supposed to tend to $p_z = \mathcal{N}(0, I)$ during the learning of the model. We could therefore use the $E(x)$ standard deviation as an anomaly score. However, because of the finite size of the training set, our experiments showed that the projection distribution of normal examples could diverge slightly from p_z. Therefore, at the end of the learning, we represent this distribution by a Gaussian distribution $\mathcal{N}(\mu, \Sigma)$ whose parameters are assessed with a validation base. The anomaly score is finally the Mahalanobis distance between $E(x)$ and μ:

$$a(x) = \sqrt{(E(x) - \mu)^T \Sigma^{-1} (E(x) - \mu)} \tag{3}$$

3.2. Theoretical analysis

In this section, we develop the loss functions L_D and L_E (see Eq. (1) and (2)) in order to identify the optimal discriminator and encoder. We begin with the discriminator. The formulas of L_D can be rewritten by introducing the integrals instead of the expectations:

$$\begin{aligned} L_D &= \int_z -p_z(z) \log D(z) dz - \int_x p_{x_n}(z) \log(1 - D(E(x))) dx - \int_x p_{x_a}(x) \log(1 - D(E(x))) dx \\ &= \int_z -p_z(z) \log D(z) - p_{z_n}(z) \log(1 - D(z)) - p_{z_a}(z) \log(1 - D(z)) dz \end{aligned}$$

where p_{z_n} (resp. p_{z_a}) is the distribution of the projection of normal (resp. anomalous) examples into the decision space. We can express the optimal discriminator D given the encoder E. Let $f(D(z))$ be the function in the integral above such that $L_D = \int_z f(D(z))dz$. To find the discriminator that minimized L_D , we set the derivative of $f(D(z))$ to 0 :

$$\begin{aligned} \frac{\partial f(D(z))}{\partial D(z)} = -\frac{p_z(z)}{D(z)} + \frac{p_n(z)}{1 - D(z)} + \frac{p_a(z)}{1 - D(z)} = 0 \\ \Rightarrow D(z) = \frac{p_z(z)}{p_z(z) + p_{z_n}(z) + p_{z_a}(z)} \end{aligned} \tag{4}$$

if $p_z(z) + p_{z_n}(z) + p_{z_a}(z) \neq 0$, we can compute the second derivative to check that this extremum is a minimum :

$$\frac{\partial f(D(z))}{\partial D(z)} = \frac{p_z(z)}{D(z)^2} + \frac{p_n(z)}{(1 - D(z))^2} + \frac{p_a(z)}{(1 - D(z))^2} > 0 \tag{5}$$

Hence the optimal discriminator is $D^*(z) = \frac{p_z(z)}{p_z(z) + p_{z_n}(z) + p_{z_a}(z)}$. It corresponds to the Bayes classifier predicting if z comes from the distribution p_z or from the encoder.

Let us see what the encoder does when the optimal discriminator is plugged into loss function L_E. We have:

$$\begin{aligned}
L_E =& \int_z p_{z_n}(z)\log(1-D^*(z)) + p_{z_a}(z)\log(D^*(z))dz \\
=& \int_z p_{z_n}(z)\log\left(\frac{p_{z_n}(z)+p_{z_a}(z)}{p_z(z)+p_{z_n}(z)+p_{z_a}(z)}\right) + p_{z_a}(z)\log\left(\frac{p_z(z)}{p_z(z)+p_{z_n}(z)+p_{z_a}(z)}\right)dz \\
=& \int_z p_{z_n}(z)\log\left(\frac{p_{z_n}(z)+p_{z_a}(z)}{p_{z_n}(z)}\frac{p_{z_n}(z)}{p_z(z)+p_{z_n}(z)+p_{z_a}(z)}\right) + p_{z_a}(z)\log\left(\frac{p_z(z)}{p_{z_a}(z)}\frac{p_{z_a}(z)}{p_z(z)+p_{z_n}(z)+p_{z_a}(z)}\right)dz \\
=& -\int_z p_{z_n}(z)\log\left(\frac{p_{z_n}(z)}{\frac{p_{z_n}(z)+p_{z_a}(z)}{2}}\right)dz - \log 2 + \int_z p_{z_n}(z)\log\left(\frac{p_{z_n}(z)}{\frac{p_z(z)+p_{z_n}(z)+p_{z_a}(z)}{3}}\right)dz - \log 3 \\
& -\int_z p_{z_a}(z)\log\left(\frac{p_{z_a}(z)}{p_z(z)}\right)dz + \int_z p_{z_a}(z)\log\left(\frac{p_{z_a}(z)}{\frac{p_z(z)+p_{z_a}(z)+p_{z_a}(z)}{3}}\right)dz - \log 3 \\
=& -KL\left(p_{z_n}\middle|\middle|\frac{p_{z_n}+p_{z_a}}{2}\right) + KL\left(p_{z_n}\middle|\middle|\frac{p_z+p_{z_n}+p_{z_a}}{3}\right) - KL\left(p_{z_a}||p_z\right) + KL\left(p_{z_a}\middle|\middle|\frac{p_z+p_{z_n}+p_{z_a}}{3}\right) \\
& -2\log 3 - \log 2
\end{aligned} \tag{6}$$

Clearly, the optimal encoder minimizes an expression composed by 4 Kullback-Leibler divergences. The minimization of the both terms $KL\left(p_{z_n}\middle|\middle|\frac{p_z+p_{z_n}+p_{z_a}}{3}\right)$ and $KL\left(p_{z_a}\middle|\middle|\frac{p_z+p_{z_n}+p_{z_a}}{3}\right)$ implies that the encoder tends to the solution where the three distribution are equal $p_z = p_{z_n} = p_{z_a}$. By maximizing $KL\left(p_{z_a}||p_z\right)$, the encoder maximizes the divergence between p_{z_a} and p_z i.e. it tries to project the anomalies outside p_z. To maximize $KL\left(p_{z_n}\middle|\middle|\frac{p_{z_n}+p_{z_a}}{2}\right)$, the encoder has to maximize the divergence between p_{z_n} and p_{z_a}, thus it aims at separating the projection of the normal examples from the projection of the anomalies. By optimizing all of these four divergences together the encoder tries to project the normal examples into p_z and to project the anomalies outside p_z and p_{x_n}.

3.3. Learning algorithm

In Algo. 1, we show the training procedure of our model. We take random examples x_n and anomalies x_a from the training set and project them into the decision space with the encoder (*i.e.*, we calculate each $E(x_a)$ and $E(x_n)$). We add random vectors from the p_z distribution to these projected examples, and get the *batch* that is presented at the input of the discriminator. The discriminator is modified by a gradient descent to minimize L_D(Eq 1); the encoder is frozen during this step. Then, it is the encoder's turn to be modified in order to minimize L_E(Eq 2), the discriminator being frozen during this step. This procedure is iterated until convergence. Notice that to seed up the gradient descent, we modify the loss function of the encoder : $L_E = -\mathbf{E}_{x_n\sim p_{x_n}}[\log(D(E(x_n)))] - \mathbf{E}_{x_a\sim p_{x_a}}[\log(1-D(E(x_a)))]$. The minimization of this function is theoretically equivalent of the previous one, but it has larger gradient for bad encoders.

Algorithm 1 AnoEAN. Adam hyper-parameters ($\alpha = 0.0002, \beta = 0.5$), Learning rate 10^{-3}

```
Require: batch size $m$, discriminator step $n_D$, $m = m_a + m_n$
Initialize the discriminator parameters $w_D$, the generator parameters $\theta_G$.
for number of training iteration do
  sample batches of $\{x_n^{(i)}\}^{m_n} \sim P_{x_n}$, $\{x_a^{(i)}\}^{m_a} \sim P_{x_a}$, $\{z^{(i)}\}^m \sim \mathcal{N}(\mu, \sigma^2)$).
  for $k = 0, ..., n_D$ do
    $\delta_w \leftarrow \nabla_w[-\frac{1}{m}\sum_{i=1}^{m} \log D_w(z^{(i)}) - \frac{1}{m_n}\sum_{i=1}^{m_n} \log(1 - D_w(E_\theta(x_n^{(i)}))) - \frac{1}{m_a}\sum_{i=1}^{m_a} \log(1 - D_w(E_\theta(x_a^{(i)})))]$
    $w_D \leftarrow Adam(\delta_w, \alpha, \beta)$
  end for
  $\delta_\theta \leftarrow \nabla_\theta[-\frac{1}{m_n}\sum_{i=1}^{m_n} \log D_w(E_\theta(x_n^{(i)})) - \frac{1}{m_a}\sum_{i=1}^{m_a} \log(1 - D_w(E_\theta(x_a^{(i)})))]$
  $\theta_G \leftarrow Adam(\delta_\theta, \alpha, \beta)$
end for
```

4. Experimentations

In this section, we describe our experimental protocol, the algorithms used to compare our model, the datasets, and the implementation details of our technique. Our results demonstrate the efficiency of our model compared to state of the art.

4.1. Experimental setup

In our experiments, we used three datasets: MNIST, KDD99, and NSL-KDD.

MNIST (LeCun, 1998) is a database of handwritten digits that is commonly used for image classification problems. Despite that MNIST does not initially contain normal classes and anomalies, it is often used in anomaly detection to analyze the behavior of algorithms and visualize their predictions. In our experiments, we used this dataset in two ways:
1) **1 against all**: we consider that a certain number represents the normal class and the remaining nine numbers represent the anomaly class. For the OCSVM, AnoGAN, ALAD, EGBAD methods, the training set consists of 5000 normal data; the test set includes 1700 examples of which 80% of normal data and 20% of anomalies randomly selected among the other nine classes. For AnoEAN and SVM, we additionally inject 10% anomalies into the learning set; these are chosen among four classes out of nine so that certain figures (anomalies) do not appear during the learning.
2) **n against m**: it is based on the same principle as the "1 against all", the only difference being that the normal class is composed of several digits. We group n classes of digits into one normal class and treat the remaining m digits as abnormal examples. We use the same sample apportionment in the learning and test sets as in "1 against all". The objective is to analyze the behavior of the algorithms in the case where the normal class is heterogeneous and can be separated into several subclasses.

The other datasets KDD99 and NSL-KDD are commonly used to evaluate the performance of anomaly detection algorithms, the objective being to detect intrusions into monitored networks. We use the same sample apportionment in the learning and test sets as before.

We compare our method with two kernel methods and three GAN-based methods.

OCSVM. The One-Class Support Vector Machine (OCSVM) is a classical kernel-based technique for novelty detection. It is usually used with the RBF or linear kernel function (Schölkopf et al., 2000). In our experiments, the RBF kernel gives the best results. The OCSVM learns a decision boundary around normal examples, containing most of the learning samples. Samples residing outside the borders are considered as abnormal. We implemented two tests, one where we feed OCSVM with normal data (fixing $\nu = 0.0001$), the other where we inject 10% anomalous example in train data and a soft margin of 10% (that is, $\nu = 0.1$). The latter did not bring any improvement to the results.

SVM. Althought the SVMs often produce efficient solutions for balanced data sets, they are sensitive to unbalanced data sets (Veropoulos et al., 1999; Wu and Chang, 2003; Akbani et al.). The main reason is that the objective function assigns the same cost C for positive and negative classification errors in terms of penalty (Veropoulos et al., 1999). We used a cost-sensitive learning solution by adding a weight for each class that penalizes errors (giving a higher weight to the least frequent class corresponding to the class of the anomalies).

AnoGAN was the first published anomaly detection method based on GAN (Schlegl et al., 2017). The model learns a generator able to project random points from a low-dimensional multivariate Gaussian distribution (latent space) to the distribution of the training data set. The model adversarially learns a discriminator that must separate the generated data from the real data. After the training stage, for each element of the test set, we sample a latent space variable using a generator, followed by gradient descent on this latent variable, to estimate the inverse projection. The anomaly criterion is a combination of the reconstruction loss and the feature match loss (distance computd with the last hidden layer of the discriminator). For each test element, if the optimization cannot find a latent variable that minimizes the criterion, we obtain a high score.

EGBAD takes advantage of the BiGAN/ALI network structure. EGBAD has an encoder that projects the data into in the latent space (Donahue et al., 2016; Dumoulin et al., 2016). The adversarial training process is driven through a discriminator that takes as an input the pair (data, latent variable) and must determine which pair constitutes a real pair consisting of a sample of real data and its coding $(x, E(x))$, or a false data sample and the corresponding latent space input of the generator $(G(z), z)$. For a given test input x, EGBAD (Zenati et al., 2018) uses the encoder to infer the latent variable $z = E(x)$ that will be used as input for the generator to reconstruct the test input $G(E(x))$. The anomaly criterion is the same as that of Schlegl et al. (2017).

ALAD is a bidirectional GAN, based on the ALICE architecture (Li et al., 2017). It directly infer the reconstruction of a data in the test phase using an encoder and the generator $G(E(x))$. In ALAD (Zenati et al.), the authors regularize the conditional distribution by adding another discriminant $D_{xx}(x, G(z))$ to approximate a conditional entropy constraint. The aim is to enhance EGBAD by explicitly forcing the encoder and generator during the training so that both are the inverse of each other. The authors also apply the spectral normalization (Miyato et al., 2018) to regularize the training. The anomaly score is the L_1 reconstruction error in the discriminant space G_{xx} between the actual data and the sample generated using the discriminant hidden layer before the logit layer.

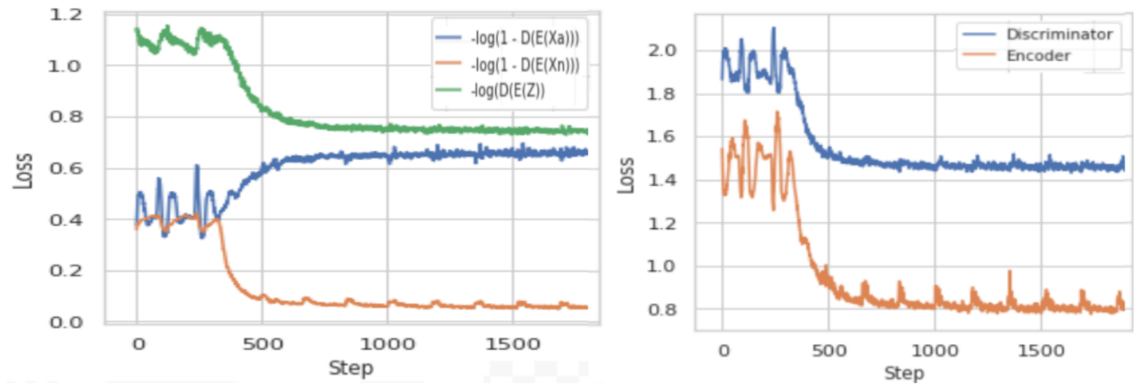


Figure 5: (a) shows the loss curves on the discriminator. (b) shows the overall loss functions of the encoder and the discriminator.

OCSVM, ALAD, AnoGAN and EGBAD are one-class methods: Only normal data is used in the learning phase. In AnoEAN and SVM, we use few additional anomalies, leading to unbalanced classes.

The performances of the algorithms are assessed with the area under the precision-recall curves (AUC). We also calculate the optimal $F_1 measure$, as well as the receiver operating characteristic(ROC curve) and associated accuracy. For each model, we select the learning parameters that maximize the AUC calculated from the anomaly score $a(x)$ Eq. (3) established with a validation set.

AnoEAN has two neural networks, the encoder and the discriminator. In the case of MNIST, the encoder is a convolutional network $(4 * 2 * 32, 4 * 2 * 64, 4 * 2 * 128, d)$(d is the dimension of latent space) and the discriminator is a multi-layer network $(32, 16, 1)$. For the NSL-KDD and KDD99 bases, the encoder and the discriminator are multi-layer networks $(128, 64, 32, d)$ and $(32, 16, 1)$ respectively. The size of the *batch* is 200. The number of *epochs* is fixed by *early stopping* and the gradient descent is performed with Adam ($lr = 10^{-3}$).

4.2. Results

In Fig. 5a, we show the curves of the loss function L_D. The green and blue curves show that the discriminator is confused through the learning steps: it cannot differentiate between the distributions over z and $E(x_n)$. Therefore, the encoder is getting better with respect to the approximation of P_z. At the same time, we see that the orange loss curve keeps decreasing, which means that the distribution of anomalous examples $E(x_a)$ diverges from P_z. In Fig. 5b, we can see that both the loss functions L_E and L_D are decreasing with the iterations until convergence.

In Fig. 6, we present our results on MNIST (1 against all): each digit is successively considered as the normal class, the others being seen as anomalies. AnoEAN outperforms all other methods in 5 out of 10 cases; its performances are comparable to the best methods on the remaining 5 digits. Notice that all the methods have poor performances on Classes 2, 3, 4 and 5 because these classes have similar visual characteristics as those of the anomalies. We

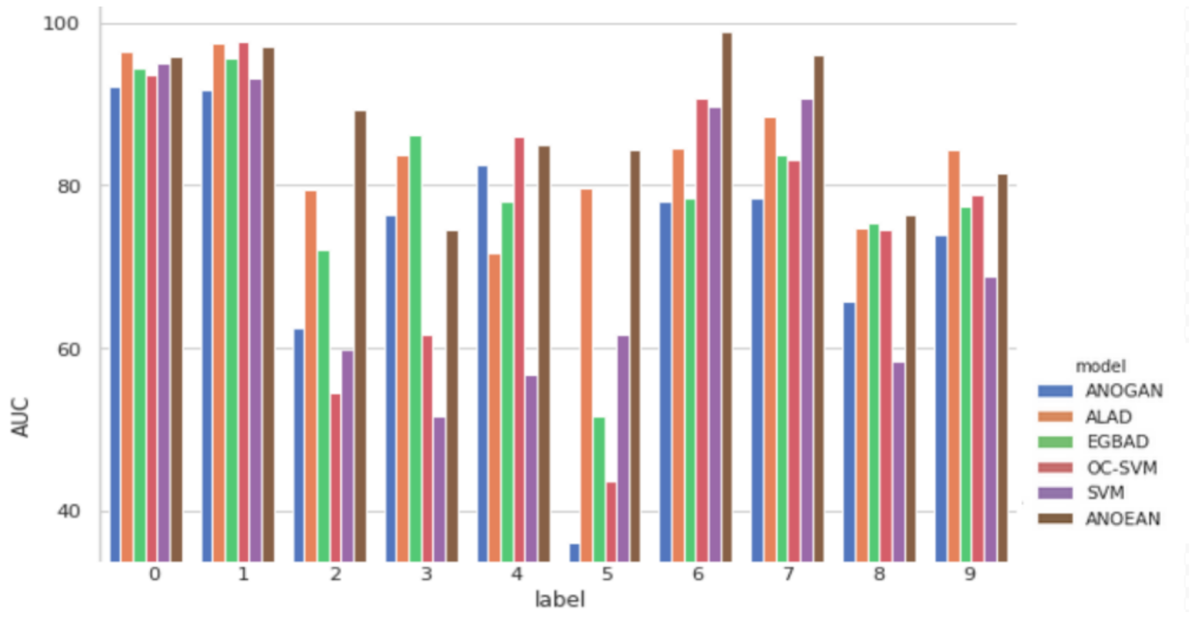


Figure 6: Model performance according to the AUC metric on each digit in the case of the MNIST problem (1 against all)

also observe that reconstruction-based approaches (ALAD and EGBAD) are competitive: They successfully rebuild the numbers, which allows them to have an important anomaly score on items that are not visually similar to the normal class.

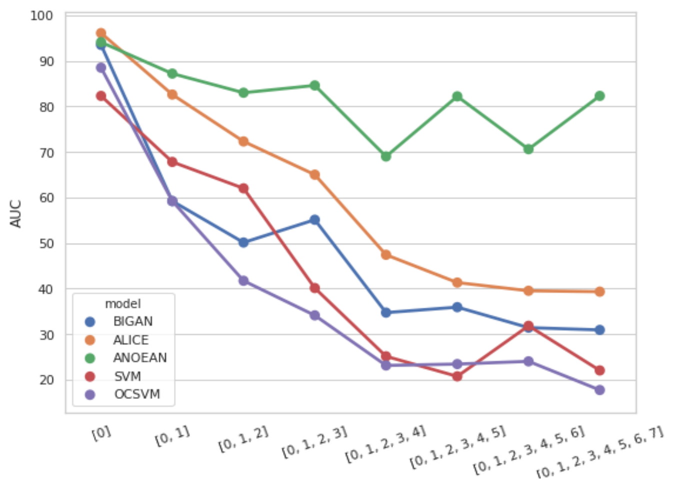


Figure 7: Visualization of the performances on different methods with several degrees of complexity in the normal class (varying the heterogeneity of the normal class with several numbers).

In Fig. 7, we show the results on MNIST (n against m). We are interested in the problem of identifying anomalies in a dataset where the normal class is heterogeneous (composed of several digits). In this configuration, AnoEAN dramatically outperforms the state of the art, even if its performance deteriorates with the increasing heterogeneity of the normal class. Notice that the performances of reconstruction-based approaches (ALAD and EGBAD) collapse. Indeed, for them to work, the generator and the encoder must necessarily be inverse from one another, what is particularly difficult to guaranty when the normal class is heterogeneous. On the other hand, AnoEAN succeeds in encoding complex distributions because it does not need to calculate the inverse of the encoder.

AUC	F1	ROC	accuracy	Model
81.3 ± 0.92	80.1±1.4	83.9±0.89	86.2±0.78	AnoGAN
95.1 ± 0.74	89.6±1.5	97.9±0.31	95.9±0.51	EGBAD
95.2 ± 0.77	90.2±0.81	98.0±0.39	96.1±0.28	ALAD
81.1 ± 4.52	78.5±4.98	91.1±2.24	88.3±7.48	OCSVM
97.9±0.26	95.2±0.63	98.8±0.28	98.3±0.25	**AnoEAN**
94.8 ± 0.5	93.9±0.8	95.6±1.49	97.4±0.24	SVM

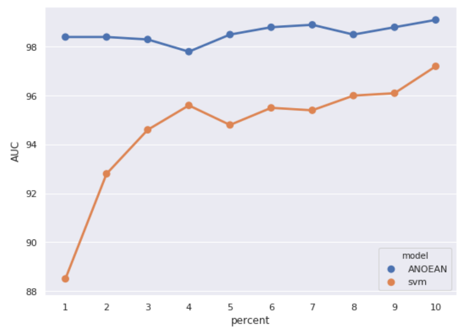
Table 1: NSL-KDD

AUC	F1	ROC	accuracy	Model
82.7±2.72	85.9±1.97	85.8±0.79	88.1±1.71	AnoGAN
95.1±3.67	96.4±3.02	98.9±0.88	98.5±1.39	EGBAD
95.6±1.51	96.8±0.72	99.1±0.37	98.7±0.28	ALAD
86.6±3.86	84.4±5.36	95±2.02	92.5±3.21	OCSVM
99.2±0.08	97.3±0.32	99.7±0.09	98.8±0.14	**AnoEAN**
98.7±0	98.5±0	98.6±0	99.3±0	SVM

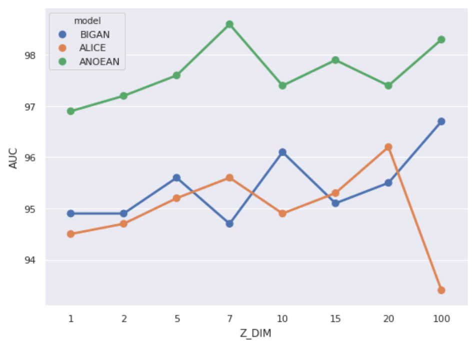
Table 2: KDD99

We present the results on NSL-KDD and KDD99 datasets in Tables 1 and 2. We obtain these results by running the train and test with ten different seeds, for the initialization of the weights and shuffling before splitting data into train and test. The values represent the mean and standard deviation confidence intervals, to show a calibrated performance of the results. These tables show several measures of performances: the area under the precision-recall curve (AUC), the F1-measure (F1), the ROC curve, and the accuracy. Since the number of anomalies is much lower than the normal examples, AUC represents our primary measure of interest. Notice that ANOEAN has a better AUC than state of the art and also a better score on the other metrics on NSL-KDD. On the KDD dataset, ANOEAN is competitive with SVM method on F1 score and accuracy. Notice that KDD dataset contains more than 400 000 single connection vectors and even with 10% of anomalies, it remains a large amount of data; This is why the SVM has good results in that case. In Fig. 8a, we reduce the amount of anomalies in the training set, and we verify that ANOEAN keeps on average the same performances, whereas those of the SVM decreases when the rate of anomalies decrease.

At last, in the Fig. 8b, we check the impact of the latent space Z dimension on the AUC. Clearly, AnoEAN performs better than the other GANs based approach (EGBAD/ALAD) whichever the dimension of Z.



(a)



(b)

Figure 8: (a) shows the impact of reducing the percentage of anomalies in AnoEAN and SVM on NSL-KDD. (b) shows the impact of the latent space vector dimension Z on the AUC metric, concerning the Adversarial Network Based models (BIGAN/ALICE/AnoEAN).

5. Conclusion and Perspectives

We presented a new anomaly detection method that uses a projection in a latent space of small dimension in which normal examples and anomalies are separated. For this, we train an encoder by *adversarial learning.* The cost function enforces the encoder learning to projects the normal data into a Gaussian distribution and the anomalies in the tail of this distribution. For the prediction, our method only needs the encoder weights. It is crucial to have a small model for applications in embedded systems where the memory is costly. We have shown that our model could discover non-encountered anomalies during the learning stage. Besides, our technique performs better than the state of the art when faced with a heterogeneous normal class. Finally, compared to state of the art, our technique triggers few false alarms. The limitation of our method is that it needs an amount of anomalous example to restrain the projection of normal examples. In future works, we will adapt our method to the case where no anomalies are available in the training set, to achieve this goal, the idea is to complete the architecture with a new generator of artificial anomalies, that should train against the encoder and the discriminator in an adversarial way.

References

Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz. Applying support vector machines to imbalanced datasets. In *Proc. ECML 2004*.

Samet Akcay, Amir Atapour Abarghouei, and Toby P. Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. *CoRR*, abs/1805.06725, 2018.

Ricardo J. G. B. Campello, Davoud Moulavi, Arthur Zimek, and Jörg Sander. Hierarchical density estimates for data clustering, visualization, and outlier detection. *TKDD*, 2015.

Raghavendra Chalapathy and Sanjay Chawla. Deep learning for anomaly detection: A survey. *CoRR*, abs/1901.03407, 2019.

Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. Anomaly detection using one-class neural networks. *CoRR*, abs/1802.06360, 2018.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):15:1–15:58, 2009. doi: 10.1145/1541880.1541882.

Antonia Creswell and Anil A. Bharath. Inverting the generator of A generative adversarial network (II). *CoRR*, abs/1802.05701, 2018.

Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, and Biswa Sengupta. Generative adversarial networks: An overview. *IEEE Signal Process.*, 2018.

Lucas Deecke, Robert A. Vandermeulen, Lukas Ruff, Stephan Mandt, and Marius Kloft. Image anomaly detection with generative adversarial networks. In *Proc. ECML 2018*.

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *CoRR*, abs/1605.09782, 2016.

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martín Arjovsky, Olivier Mastropietro, and Aaron C. Courville. Adversarially learned inference. *CoRR*, abs/1606.00704, 2016.

Olga Galinina, Sergey Andreev, Sergey I. Balandin, and Yevgeni Koucheryavy, editors. *State of the Art Literature Review on Network Anomaly Detection with Deep Learning*, volume 11118 of *Lecture Notes in Computer Science*, 2018.

Izhak Golan and Ran El-Yaniv. Deep anomaly detection using geometric transformations. In *Proc. NIPS 2018*, pages 9781–9791, 2018.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*. 2014a.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014b.

Victoria J. Hodge and Jim Austin. A survey of outlier detection methodologies. *Artif. Intell. Rev.*, 22(2):85–126, 2004. doi: 10.1023/B:AIRE.0000045502.10941.a9.

B. Ravi Kiran, Dilip Mathew Thomas, and Ranjith Parakkal. An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos. 2018.

Y. LeCun. The MNIST database of handwritten digits. 1998.

Chunyuan Li, Hao Liu, Changyou Chen, Yunchen Pu, Liqun Chen, Ricardo Henao, and Lawrence Carin. ALICE: towards understanding adversarial learning for joint distribution matching. In *Proc. NIPS 2017*, pages 5501–5509, 2017.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *CoRR*, abs/1802.05957, 2018.

Marco A. F. Pimentel, David A. Clifton, Lei A. Clifton, and Lionel Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014. doi: 10.1016/j.sigpro.2013.12.026.

Lukas Ruff, Nico Görnitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Robert A. Vandermeulen, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *ICML*, 2018.

Mohammad Sabokrou, Mohammad Khalooei, Mahmood Fathy, and Ehsan Adeli. Adversarially learned one-class classifier for novelty detection. In *Proc. CVPR 2018*.

Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems 29*, 2016.

Thomas Schlegl, Philipp Seeböck, Sebastian M. Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. *CoRR*, abs/1703.05921, 2017.

B. Schölkopf, R.C. Williamson, A.J. Smola, J. Shawe-Taylor, and J. Platt. Support vector method for novelty detection. *Advances in neural information processing systems*, 2000.

K. Veropoulos, C. Campbell, and N. Cristianini. Controlling the sensitivity of support vector machines. *Proc. IJCAI 1999*, 1999. doi: abs/1802.05957.

Gang Wu and Edward Y. Chang. Adaptive feature-space conformal transformation for imbalanced-data learning. In *Proc. ICML 2003*, pages 816–823, 2003.

Yan Xia, Xudong Cao, Fang Wen, Gang Hua, and Jian Sun. Learning discriminative reconstructions for unsupervised outlier removal. In *Proc. 2015 IEEE International Conference on Computer Vision (ICCV'15)*, pages 1511–1519, 2015. doi: 10.1109/ICCV.2015.177.

Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Yang Feng, Jie Chen, Zhaogang Wang, and Honglin Qiao. Unsupervised anomaly detection via variational auto-encoder for seasonal KPIs in web applications. *CoRR*, 2018.

Houssam Zenati, Manon Romain, Chuan-Sheng Foo, Bruno Lecouat, and Vijay Chandrasekhar. Adversarially learned anomaly detection. In *Proc. ICDM 2018*.

Houssam Zenati, Chuan Sheng Foo, Bruno Lecouat, Gaurav Manek, and Vijay Ramaseshan Chandrasekhar. Efficient GAN-based anomaly detection. *CoRR*, 2018.

Wei Zhao, Pengpeng Yang, Rongrong Ni, Yao Zhao, and Wenjie Li. Cycle gan-based attack on recaptured images to fool both human and machine. In *17th International Workshop, IWDW 2018*.