Adversarially Learned Anomaly Detection

Houssam Zenati*†¶, Manon Romain*‡¶, Chuan-Sheng Foo*¶, Bruno Lecouat§¶ and Vijay Chandrasekhar¶
†*CentraleSupélec, houssam.zenati@student.ecp.fr*
‡*École Polytechnique, manon.romain@polytechnique.edu*
§*Télécom ParisTech*
¶*Institute for Infocomm Research, A*STAR, {foocs,bruno_lecouat,vijay}@i2r.a-star.edu.sg*
**Authors contributed equally to this work*

***Abstract*—Anomaly detection is a significant and hence well-studied problem. However, developing effective anomaly detection methods for complex and high-dimensional data remains a challenge. As Generative Adversarial Networks (GANs) are able to model the complex high-dimensional distributions of real-world data, they offer a promising approach to address this challenge. In this work, we propose an anomaly detection method, Adversarially Learned Anomaly Detection (ALAD) based on bi-directional GANs, that derives adversarially learned features for the anomaly detection task. ALAD then uses reconstruction errors based on these adversarially learned features to determine if a data sample is anomalous. ALAD builds on recent advances to ensure data-space and latent-space cycle-consistencies and stabilize GAN training, which results in significantly improved anomaly detection performance. ALAD achieves state-of-the-art performance on a range of image and tabular datasets while being several hundred-fold faster at test time than the only published GAN-based method.**

***Keywords*-Anomaly Detection, Unsupervised Learning, Deep Learning, Generative Adversarial Networks, Novelty Detection**

I. Introduction

Anomaly detection is a problem of great practical significance across a range of real-world settings, including cyber-security [1], manufacturing [2], fraud detection, and medical imaging [3]. Fundamentally, anomaly detection methods need to model the patterns in normal data to identify atypical samples. Although anomaly detection is a well-studied problem [3]–[5], developing effective methods for complex and high-dimensional data remains a challenge.

Generative Adversarial Networks (GANs) [6] are a powerful modeling framework for high-dimensional data that could address this challenge. Standard GANs consist of two competing networks, a generator G and discriminator D. G models the data by learning a mapping from latent random variables z (drawn from Gaussian or uniform distributions) to the data space, while D learns to distinguish between real data and samples generated by G. GANs have been empirically successful as a model for natural images [7], [8] and are increasingly being used in speech [9] and medical imaging applications. For example, [10] proposes a method that uses a standard GAN for anomaly detection on eye images. However, at test time, the method requires solving an optimization problem for each example to find a latent z such that $G(z)$ yields a visually similar image that is also on the image manifold modeled by G; this z is then used to compute an anomaly score for the example. The need to solve an optimization problem for every test example makes this method impractical on large datasets or for real-time applications.

In this work, we propose a GAN-based anomaly detection method that is not only effective, but also efficient at test time. Specifically, our method utilizes a class of GANs that simultaneously learn an encoder network during training [11], [12], thus enabling faster and more efficient inference at test time than [10]. In addition, we incorporate recent techniques to further improve the encoder network [13] and stabilize GAN training [14], and show through ablation studies that these techniques also improve performance on the anomaly detection task. Experiments on a range of high-dimensional tabular and image data demonstrate the efficiency and effectiveness of our approach.

II. Related Work

Anomaly detection has been extensively studied, as surveyed in [3]–[5]. As such, here we give a brief overview and refer the reader to these reviews for a more in-depth discussion. A major class of classic anomaly detection methods are distance-based, using distances to nearest neighbors or clusters in the data to assess if data is anomalous. Such methods rely on having an appropriate distance metric for the data. One-class classification approaches trained only on normal data such as one-class SVMs [15] are also widely used; these methods learn a classification boundary around the normal data. A third class of methods uses fidelity of reconstruction [5] to determine whether an example is anomalous. Principal component analysis and variants thereof [16]–[18] are examples of such methods.

More recent works are based on deep neural networks; we note that neural networks have a long history of being used for anomaly detection [19]. Approaches based on auto-encoders [20] and variational auto-encoders [21] first train a

Proceedings of the 20th IEEE International Conference on Data Mining, Singapore, 2018. All code and hyperparameters can be found at https://github.com/houssamzenati/Adversarially-Learned-Anomaly-Detection

model to reconstruct normal data and subsequently identify anomalies as samples with high reconstruction errors. Energy based models [22] and deep auto-encoding Gaussian mixture models [23] have also been explored specifically for the purpose of anomaly detection. Such methods model the data distribution using auto-encoders or similar models, and derive statistical anomaly criterion based on energies or mixtures of Gaussians. Finally, GANs have been applied to anomaly detection in the context of medical imaging on retinal images [10]. However, the methods proposed in [10] require an inference procedure at test time, where latent variables z are recovered using stochastic gradient descent for every test example. This inference procedure is computationally expensive as every gradient computation requires backpropagation through the generator network.

Our proposed ALAD method is most closely related to the AnoGAN method described in [10]. However, in contrast to AnoGAN, which uses a standard GAN, ALAD builds upon bi-directional GANs, and hence also includes an encoder network that maps data samples to latent variables. This enables ALAD to avoid the computationally expensive inference procedure required by AnoGAN since the required latent variables can be recovered using a single feed-forward pass through the encoder network at test time. Our anomaly scoring criteria is different from AnoGAN, and we also incorporate recent advances to stabilize the GAN training procedure in ALAD.

III. Background

Standard GANs consist of two neural networks, a *generator* G and *discriminator* D, and are trained on a set of M unlabeled data samples $\left\{x^{(i)}\right\}_{i=1}^{M}$. The generator G maps random variables z sampled from a latent distribution (typically Gaussian or uniform) to the input data space. The discriminator D tries to distinguish real data samples $x^{(i)}$ from samples $G(z)$ generated by G. Informally, these two networks G and D are in competition – G tries to generate samples that resemble real data, while D attempts to discriminate between samples produced by the generator and real data samples. Training a GAN then typically involves taking alternating gradient steps so that G can better "fool" D, and D can better detect samples from G.

Formally, define $p_{\mathcal{X}}(x)$ to be the distribution over data x in the data space $\mathcal{X}$, and $p_{\mathcal{Z}}(z)$ the distribution over latent generator variables z in the latent space $\mathcal{Z}$. Then training a GAN involves finding the discriminator D and the generator G that solve the saddle-point problem $\min_G \max_D V(D, G)$ where

$$\begin{aligned} V(D,G) = {} & \mathbb{E}_{x \sim p_{\mathcal{X}}}\left[\log D(x)\right] \\ & + \mathbb{E}_{z \sim p_{\mathcal{Z}}}\left[\log\left(1 - D\left(G(z)\right)\right)\right]. \end{aligned}$$

The solutions to this saddle-point problem are described by Lemma III.1 (proved in [6]), which shows that the optimal generator induces a distribution $p_G(x)$ that matches the true data distribution $p_{\mathcal{X}}(x)$.

Lemma III.1. *For G fixed, the optimal discriminator D^*_G is:*

$$D^*_G = \frac{p_{\mathcal{X}}(x)}{p_{\mathcal{X}}(x) + p_G(x)}.$$

And for this optimal discriminator D^*_G the global minimum of the virtual training criterion $C(G) = \max_D V(D, G)$ is achieved if and only if $p_G(x) = p_{\mathcal{X}}(x)$.

D and G are typically determined via alternating gradient descent on the parameters of D and G, treating the other as fixed, to maximize (for D) or minimize (for G) $V(D, G)$ accordingly. Having trained the GAN, one can approximately sample from $p_{\mathcal{X}}$ using the generator taking $G(z)$ with $z \sim p_{\mathcal{Z}}$. As explained in Section IV-C being able to learn the distribution of the normal data is key for the anomaly detection task. Note that it is not possible to explicitly compute a likelihood or obtain the latent representation for a given data sample x directly using the GAN.

IV. Adversarially Learned Anomaly Detection

Given that standard GANs only support efficient sampling, there are several approaches one can take in order to adapt them for anomaly detection. For instance, for a data point x, one could use sampling [24] to estimate the likelihood of x and determine if it is an outlier. However, while sampling from a GAN is efficient, accurate estimation of likelihoods typically requires a large number of samples, thus making the likelihood computation computationally expensive. Another approach is to "invert" the generator to find latent variables z that minimize reconstruction error or related objectives by stochastic gradient descent [10], [25]–[27]. This approach is also computationally costly as each gradient computation requires backpropagation through the generator network.

A. GAN architecture

Motivated by computational efficiency, we instead build on a class of GANs that simultaneously learns an encoder network E which maps data samples x to the latent space z during training [11], [12]. Computing a (approximate) latent representation for a data point x in such models is done simply by passing x through the encoder network. Our models also incorporate recent improvements to improve the encoder network [13] by adding an additional discriminator to encourage cycle-consistency, *i.e.* that $G(E(x)) \approx x$.

Formally, the BiGAN [12] and AliGAN [11] models match the joint distribution $p_G(x, z) = p_{\mathcal{Z}}(z) p_G(x|z)$ and $p_E(x, z) = p_{\mathcal{X}}(x) p_E(z|x)$ with an adversarial discriminator network D_{xz} that takes x and z as inputs. BiGAN and AliGAN determine the discriminator D_{xz}, the generator G and the encoder E as the solution to the saddle-point problem:

$\min_{G,E} \max_{D_{xz}} V(D_{xz}, E, G)$, with $V(D_{xz}, E, G)$ defined as:

$$\begin{aligned} V(D_{xz}, E, G) = \mathbb{E}_{x \sim p_{\mathcal{X}}} [\log D_{xz}(x, E(x))] \\ + \mathbb{E}_{z \sim p_{\mathcal{Z}}} [1 - \log D_{xz}(G(z), z)] \end{aligned}$$

The solutions of the saddle-point problem and the distribution matching property $p_E(x, z) = p_G(x, z)$ are described by Lemma IV.1 [11], [12]:

Lemma IV.1. *For E and G fixed, the optimal discriminator $D^*_{xz,E,G}$ is:*

$$D^*_{xz} = \frac{p_E(x, z)}{p_E(x, z) + p_G(x, z)}.$$

And for this optimal discriminator D^*_{xz} the global minimum of the virtual training criterion $C(E, G) = \max_{D_{xz}} V(D_{xz}, E, G)$ is achieved if and only if $p_E(x, z) = p_G(x, z)$.

While in theory the joint distributions $p_E(x, z)$ and $p_G(x, z)$ will be identical, in practice this is often not the case as training does not necessarily converge to a solution of the saddle-point problem. This potentially results in a violation of cycle-consistency so $G(E(x)) \not\approx x$ as highlighted in [13]; such a violation would create issues for reconstruction-based anomaly detection methods.

To solve this problem, the ALICE framework [13] proposes to approximate the conditional entropy $H^\pi(x|z) = -\mathbb{E}_{\pi(x,z)} [\log \pi(x|z)]$ (where $\pi(x, z)$ is a joint distribution over x and z) in an adversarial manner to encourage cycle consistency. This saddle-point problem $\min_{G,E} \max_{D_{xz}} V_{ALICE}(D_{xz}, E, G)$ includes the conditional entropy regularization (V_{CE}) on the encoder E and the generator G:

$$V_{ALICE}(D_{xz}, E, G) = V(D_{xz}, E, G) + V_{CE}(E, G)$$

The conditional entropy regularization imposed on the encoder E and the generator G can be approximated with an additional discriminator network $D_{xx}(x, \hat{x})$

$$\begin{aligned} V(D_{xx}, E, G) = \mathbb{E}_{x \sim p_{\mathcal{X}}} [\log D_{xx}(x, x)] \\ + \mathbb{E}_{x \sim p_{\mathcal{X}}} [1 - \log D_{xx}(x, G(E(x)))] \end{aligned}$$

and [13] formally show that such a discriminator will indeed enforce cycle consistency.

B. Stabilizing GAN training

To stabilize the training of the baseline ALICE model, we further regularize the conditional distributions by adding another conditional entropy constraint, and apply spectral normalization [14].

Formally, we regularize the latent space conditional $H^\pi(z|x) = -\mathbb{E}_{\pi(x,z)} [\log \pi(z|x)]$ (where $\pi(x, z)$ is a joint distribution over x and z) with an additional adversarially learned discriminator D_{zz}, with a similar saddle-point objective as follows; the proof of matching of conditionals is a simple adaptation of those presented in [13].

$$\begin{aligned} V(D_{zz}, E, G) = \mathbb{E}_{z \sim p_{\mathcal{Z}}} [\log D_{zz}(z, z)] \\ + \mathbb{E}_{z \sim p_{\mathcal{Z}}} [1 - \log D_{zz}(z, E(G(z)))] \end{aligned}$$

Putting it all together, our Adversarially Learned Anomaly Detection (ALAD) method solves the following saddle-point problem during training: $\min_{G,E} \max_{D_{xz}, D_{xx}, D_{zz}} V(D_{xz}, D_{xx}, D_{zz}, E, G)$, with $V(D_{xz}, D_{xx}, D_{zz}, E, G)$ defined as

$$\begin{aligned} V(D_{xz}, D_{xx}, D_{zz}, E, G) = \\ V(D_{xz}, E, G) + V(D_{xx}, E, G) + V(D_{zz}, E, G). \end{aligned}$$

A schematic of this final GAN model is shown in Figure 1.

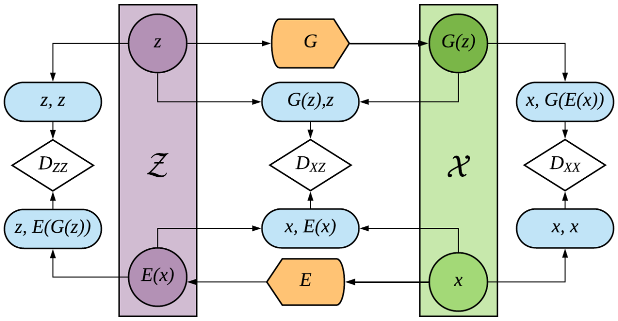

Figure 1. The GAN used in Adversarially Learned Anomaly Detection. D_{zz}, D_{xz} and D_{xx} denote discriminators (white), G the generator (orange), and E the encoder (orange); these networks are simultaneously learned during training.

Our addition of spectral normalization is motivated by recent work [14], [28]–[30] that shows the efficiency of adding Lipschitz constraints on the discriminators of GANs to stabilize the training. In particular [14] proposes a simple re-parametrization of the weights which turns out to be very effective in practice. They propose to fix the spectral norm of the weight matrix (*i.e.,* its largest eigenvalues) of hidden layers in the discriminator. This method is computationally efficient and stabilizes training. In our experiments, we found that spectral normalization was also beneficial when employed to regularize the encoder (as opposed to the discriminator alone). Note that the original ALICE models [13] did not include this re-parametrization of the weights that we included in our models.

C. Detecting anomalies

ALAD is a reconstruction-based anomaly detection technique that evaluates how far a sample is from its reconstruction by the GAN. Normal samples should be accurately reconstructed whereas anomalous samples will likely be poorly reconstructed. This intuition is illustrated in Figure 2.

To this end, we first need to effectively model the data distribution: this is achieved using the described GAN, where the generator G is used to learn the distribution of the normal data so that $p_G(x) = p_{\mathcal{X}}(x)$, where $p_G(x) = \int p_G(x|z)p_{\mathcal{Z}}(z)dz$. We also need to learn the manifold of the data so as to recover precise latent representations that result in faithful reconstructions of normal samples; the two symmetric conditional entropy cycle-consistency regularization terms in our model help ensure this.

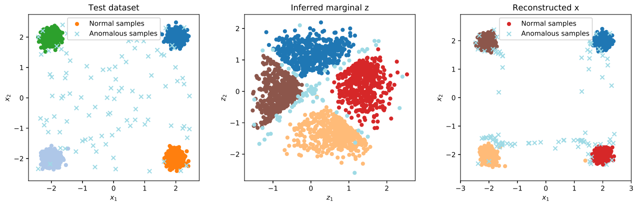

Figure 2. Toy data example with anomalous samples, encoded representations and reconstructions - circles are normal samples while crosses are anomalous samples.

The other key component of ALAD is an anomaly score that quantifies the distance between the original samples and their reconstructions. The obvious choice of Euclidean distance between the original samples and their reconstructions in data space may not be a reliable measure of dissimilarity. For instance, in the case of images, it can be very noisy because images with similar visual features are not necessarily close to each other in terms of Euclidean distance.

We instead propose to compute the distance between samples in the feature space of the cycle-consistency discriminator D_{xx}, as defined by the layer before the logits; these features are also known as CNN codes. Concretely, having trained a model on the normal data to provide E, G, D_{xz}, D_{xx} and D_{zz}, we define a score function $A(x)$ based on the L_1 reconstruction error in this feature space:

$$A(x) = \left|\left|f_{xx}(x,x) - f_{xx}(x, G(E(x)))\right|\right|_1.$$

Here, $f(\cdot,\cdot)$ denotes the activations of the layer before the logits (CNN codes) in the D_{xx} network for the given pair of input samples. $A(x)$ captures the discriminators' confidence that a sample is well encoded and reconstructed by our generator and therefore derived from the real data distribution. Samples with larger values of $A(x)$ are deemed more likely to be anomalous. The procedure for computing $A(x)$ is described in Algorithm 1. Our analyses in Section V-F confirm the effectiveness of our proposed anomaly score compared to other possible variants.

The proposed anomaly score is inspired by the feature-matching loss presented in [31]. However, unlike [31] where the feature-matching loss is based on features computed with the discriminator of a standard GAN (that distinguishes between generated samples and real data), the features used by ALAD are computed using the D_{xx} discriminator (that takes pairs of data as input) which does not exist in the standard GAN model considered in [31]. In addition, we do not use the feature-matching loss within the GAN training procedure, but instead use the concept in a different context of computing an anomaly score at inference time.

Algorithm 1 Adversarially Learned Anomaly Detection

```
Input  $\boldsymbol{x}, \sim p_{\mathcal{X}_{Test}}(x), E, G, f_{xx}$ where $f_{xx}$ is the feature layer of $D_{xx}$
Output $A(\boldsymbol{x})$, where $A$ is the anomaly score
procedure INFERENCE
    $\tilde{\boldsymbol{z}} \leftarrow E(\boldsymbol{x})$                    ▷ Encode samples
    $\hat{\boldsymbol{x}} \leftarrow G(\tilde{\boldsymbol{z}})$,                  ▷ Reconstruct samples
    $f_\delta \leftarrow f_{xx}(\boldsymbol{x}, \hat{\boldsymbol{x}})$
    $f_\alpha \leftarrow f_{xx}(\boldsymbol{x}, \boldsymbol{x})$
    return $\|f_\delta - f_\alpha\|_1$
end procedure
```

Why not use the output of the D_{xx} discriminator?

We propose that feature loss computed using D_{xx} is preferable over the output of the model D_{xx} for the anomaly score. To see why, consider that D_{xx} is supposed to differentiate between a real sample pair (x, x) and its reconstruction $(x, G(E(x))$. However, at a solution to the GAN saddle-point problem the generator and encoder will perfectly capture the true data and latent variable distribution. In this case, D_{xx} will be unable to discriminate between the real and reconstructed samples, and thus will output a random prediction that would not be an informative anomaly score. Our analyses in Section V-F confirm that our proposed anomaly score performs better than the output of D_{xx}.

V. Experiments

A. *Experimental setup*

Datasets: We evaluated our Adversarially Learned Anomaly Detection method on publicly available tabular and image datasets (Table I). For tabular data, we used the KDDCup99 10% dataset [32] as well as the Arrhythmia dataset [32]. The KDDCup99 10% dataset (that we denote by KDD99) is a network intrusion dataset, while the Arrhythmia dataset consists of data in the presence and absence of cardiac arrhythmia as well as classifications into 1 of 16 groups. For the image datasets, we used the SVHN dataset [33] containing images of house numbers, and the CIFAR-10 dataset [34] that contains 10 classes of images.

Table I
Statistics of the public benchmark datasets

Dataset	Features	Total Instances
KDD99	121	494021
Arrhythmia	274	452
SVHN	3072	99289
CIFAR-10	3072	60000

Data setup and evaluation metrics: For the tabular data, we follow the experimental setups of [22], [23]. Due to the high proportion of outliers in the KDD dataset, "normal" data are treated as anomalies and the 20% of samples with the highest anomaly scores $A(x)$ are classified as anomalies (positive class). For the arrhythmia dataset, anomalous classes represent 15% of the data and therefore the 15% of samples with the highest anomaly scores are likewise classified as anomalies (positive class). We evaluate our model with the same metrics (Precision, Recall, F1 score) as the state-of-the-art deep learning baselines [22], [23]. For the image data, we generate ten different datasets each from SVHN [33] and CIFAR-10 [34] by successively treating images from one class as normal and considering images from the remaining 9 as anomalous. For each dataset, we use 80% of the whole official dataset for training and keep the remaining 20% as our test set. We further remove 25% from the training set for a validation set and discard anomalous samples from both training and validation sets (thus setting up a novelty detection task). We evaluate models using the area under the receiver operating curve (AUROC). Further details on the experimental setup are provided in the Appendix.

B. Baselines

We compare ALAD against several anomaly detection methods, which we briefly describe below.

One Class Support Vector Machines (OC-SVM) [15] are a classic kernel method for novelty detection that learns a decision boundary around normal examples. We use the radial basis function kernel in our experiments. The ν parameter is set to the expected anomaly proportion in the dataset, which is assumed to be known, whereas the γ parameter is set to $1/m$ where m is the number of input features.

Isolation Forests (IF) [35] are a newer classic machine learning technique that isolates anomalies instead of modeling the distribution of normal data. The method proceeds by first building trees using randomly selected split values across randomly chosen features. Then, the anomaly score is defined to be the average path length from a particular sample to the root. We use default parameters provided by the scikit-learn [36] package in our experiments.

Deep Structured Energy Based Models (DSEBM) [22] are a state-of-the-art method that utilize energy-based models. The main idea is to accumulate the energy across layers that are used similarly to those in a denoising autoencoder. In the original paper, two anomaly scoring criteria were investigated: energy and reconstruction error. We include both criteria in our experiments as DSEBM-r (reconstruction) and DSEBM-e (energy). Model details are provided in Appendix A1.

Deep Autoencoding Gaussian Mixture Model (DAGMM) [23] is a state-of-the-art autoencoder-based method for anomaly detection. The method first jointly trains an autoencoder to generate a sensible latent space and reconstruction features, as well as an estimator network that will output parameters of a GMM modeling this lower-dimensional latent space. At test time, the likelihood of a sample's latent and reconstruction features as determined using the learned GMM is used as the anomaly detection metric.

AnoGAN [10] is the only published GAN-based method for anomaly detection. The method involves training a DCGAN [7], and at inference time using it to recover a latent representation for each test data sample. The anomaly criterion is a combination of reconstruction and discrimination components. The reconstruction component measures how well the GAN is able to reconstruct the data via the generator, while the discrimination component considers a score based on the discriminator. The original paper [10] compares two approaches for the anomaly score and we picked the variant which performed best in the paper. For tabular data, we adapted the GAN by using fully connected layers.

C. Experiments on tabular data

We report in Table II results on KDD99 [32] as well as Arrhythmia [32] where we see that ALAD is competitive with state-of-the-art methods. Results for OC-SVM, DSEBM, DAGMM were obtained from [22], [23] while results for other baselines as well as ALAD are averages over 10 runs.

Table II
PERFORMANCE ON TABULAR DATASETS

Dataset	Model	Precision	Recall	F1 score
KDD99	OC-SVM	0.7457	0.8523	0.7954
	IF	0.9216	0.9373	0.9294
	DSEBM-r	0.8521	0.6472	0.7328
	DSEBM-e	0.8619	0.6446	0.7399
	DAGMM	0.9297	0.9442	0.9369
	AnoGAN	0.8786	0.8297	0.8865
	ALAD	**0.9427**	**0.9577**	**0.9501**
Arrhythmia	OC-SVM	**0.5397**	0.4082	0.4581
	IF	0.5147	**0.5469**	**0.5303**
	DSEBM-r	0.1515	0.1513	0.1510
	DSEBM-e	0.4667	0.4565	0.4601
	DAGMM	0.4909	0.5078	0.4983
	AnoGAN	0.4118	0.4375	0.4242
	ALAD	0.5000	0.5313	0.5152

Our ALAD method outperforms DAGMM, the best deep learning based method. Interestingly, we observe that Isolation Forests is competitive on the KDD99 dataset and achieves state-of-the-art results on the small Arrhythmia dataset. The lack of sufficient training data could have resulted in poorer performance of the data hungry deep learning based methods. On the large KDD99 dataset however, ALAD significantly outperforms all other methods.

D. Experiments on image data

We also evaluate our model on SVHN [33] as well as CIFAR-10 [32]. We report the results for individual tasks (Figures 3 and 4) and the average performance over all tasks over 3 runs (Table III).

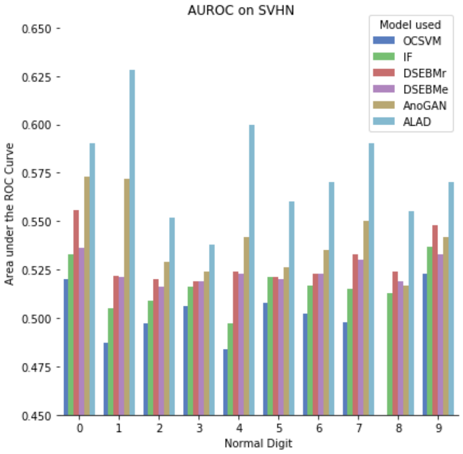

Figure 3. Area under the ROC curve for the SVHN dataset

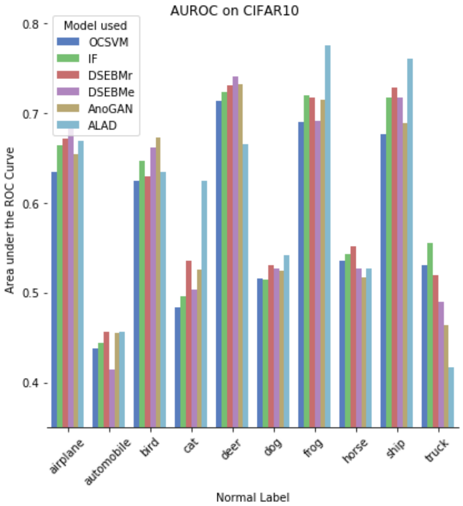

Figure 4. Area under the ROC curve for the CIFAR-10 dataset

Table III
PERFORMANCE ON IMAGE DATASETS

Dataset	Model	AUROC
SVHN	OC-SVM	0.5027 ± 0.0132
	IF	0.5163 ± 0.0120
	DSEBM-r	0.5290 ± 0.0129
	DSEBM-e	0.5240 ± 0.0067
	AnoGAN	0.5410 ± 0.0193
	ALAD	**0.5753 ± 0.0268**
CIFAR-10	OC-SVM	0.5843 ± 0.0956
	IF	0.6025 ± 0.1040
	DSEBM-r	0.6071 ± 0.1007
	DSEBM-e	0.5956 ± 0.1151
	AnoGAN	0.5949 ± 0.1076
	ALAD	**0.6072 ± 0.1201**

We observe that on the SVHN dataset, ALAD outperforms all other methods, while on the CIFAR-10 dataset, ALAD outperforms all other methods on 5 of the 10 tasks, and is competitive on the remaining 5. We also observe that there are some tasks where ALAD does not perform well. On the SVHN dataset, ALAD performs badly on the task where “3” is the normal digit. This is possibly because “3” is visually similar to several other digits like “2” and “5” such that the model can produce a relatively good reconstruction even for images from these anomalous digit classes (Figure 5; rows 3, 4), in spite of the fact that these anomalous images tend to be reconstructed to look like the digit “3” (that the model was trained on). We observe similar behavior on the CIFAR-10 dataset when we consider the task where “Automobile” is the normal class, a task where ALAD performs badly. We see that anomalous samples from the “Truck” or “Ship” classes are also reconstructed relatively well, even though the reconstructions also look like cars (Figure 6; rows 3, 4). As a sanity check, we see from the first two rows of Figures 5 and 6 that the ALAD model is able to reconstruct examples from the normal class reasonably well.

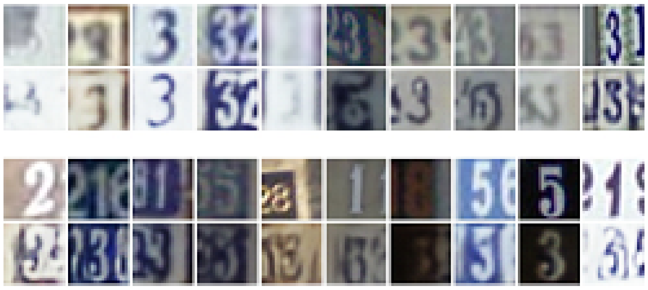

Figure 5. Error analysis on SVHN when the normal class is the “3” digit. 1st row: normal data; 2nd row: reconstruction of normal data; 3rd row: anomalous data; 4th row: reconstruction of anomalous data.

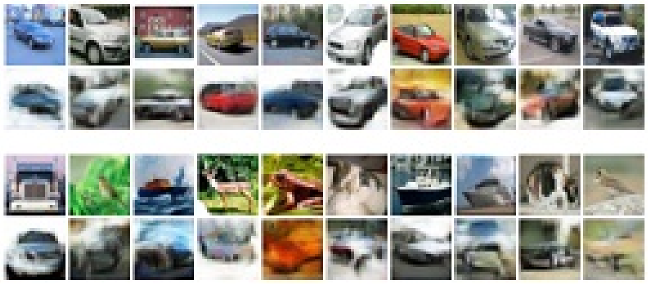

Figure 6. Error analysis on CIFAR-10 when the normal class is “Automobile”. 1st row: normal data; 2nd row: reconstruction of normal data; 3rd row: anomalous data; 4th row: reconstruction of anomalous data.

We also compared the inference time of ALAD and AnoGAN [10], the only other GAN-based anomaly detection method (Table IV). On both SVHN [33] and CIFAR-10 [34] datasets, the inference time is reported for the first task. We see that ALAD is orders of magnitude faster than AnoGAN.

Table IV
AVERAGE INFERENCE TIME (MS) ON A GEFORCE GTX TITAN X

Dataset	Batch Size	AnoGAN	ALAD	Speed Up
KDD99	50	1235	1.4	$\sim$ 900
Arrhythmia	32	1102	41	$\sim$ 30
SVHN	32	10496	10.5	$\sim$ 1000
CIFAR-10	32	10774	10.5	$\sim$ 1000

E. Ablation studies

Here we demonstrate the utility of the additional components we used to stabilize the basic ALICE model (Section IV-B) by systematically removing each component in turn. Specifically, we repeated our experiments with and without spectral normalization and the additional conditional entropy regularization $H^{\pi}(z|x)$. Our full ALAD model includes both SN (spectral norm) and DL (discriminator in latent space).

Table V
ABLATION STUDY ON KDD99 AND ARRHYTHMIA DATASETS

Model	**Precision**	**Recall**	**F1 score**
	KDD99		
Baseline	0.948±0.007	0.963±0.007	0.955±0.007
Baseline + DL	0.944±0.008	0.959±0.008	0.951±0.008
Baseline + SN	0.942±0.004	0.957±0.004	0.949±0.004
Baseline + SN + DL	0.943±0.002	0.958±0.002	0.950±0.002
	Arrhythmia		
Baseline	0.477±0.039	0.506±0.041	0.491±0.040
Baseline + DL	0.497±0.040	0.528±0.043	0.512±0.042
Baseline + SN	0.500±0.021	0.531±0.022	0.515±0.021
Baseline + SN + DL	0.482±0.035	0.513±0.037	0.497±0.036

The results in Table V show that adding the latent penalization improves results on Arrhythmia while it does not affect performance on KDD99. For SVHN and CIFAR-10, we show results averaged over all tasks in Table VI; task specific results are shown in the Appendix (Figures 7, 8). On the SVHN tasks, adding the spectral norm (SN) and the discriminator in latent space (DL) improves performance but they had minimal effect on the CIFAR-10 tasks.

Table VI
ABLATION STUDY ON SVHN AND CIFAR-10

Model	**AUROC**
SVHN	
Baseline	0.5194 ± 0.0371
Baseline + DL	0.5289 ± 0.0384
Baseline + SN	0.5513 ± 0.0357
Baseline + SN + DL	0.5753 ± 0.0267
CIFAR-10	
Baseline	0.5701 ± 0.1282
Baseline + DL	0.5361 ± 0.1348
Baseline + SN	0.5991 ± 0.1308
Baseline + SN + DL	0.6072 ± 0.1201

F. Exploration of different anomaly scores

In this section we evaluate our proposed anomaly score and compare it to other possibilities for reconstruction-based criteria. In particular, we consider the raw output of the discriminator (to which we apply a *log*) referred to as "Logits" but also using non adversarially learned L_1 and L_2 reconstruction errors. Our approach is denoted by "Features", as it uses features computed by the discriminator. Formally, the scores are defined below, where x is the input sample and $x' = G(E(x))$ is its reconstruction through ALAD:

- L_1 : $A(x) = ||x - x'||_1$
- L_2 : $A(x) = ||x - x'||_2$
- Logits : $A(x) = \log(D_{xx}(x, x'))$
- Features : $A(x) = ||f_{xx}(x, x) - f_{xx}(x, x')||_1$

Table VII
DIFFERENT SCORE METHODS ON KDD99 AND ARRHYTHMIA DATASETS

Model	**Precision**	**Recall**	**F1 score**
	KDD99		
L_1	0.9113 ± 0.0627	0.9258 ± 0.0637	0.9185 ± 0.0632
L_2	0.9316 ± 0.0155	0.9464 ± 0.0157	0.9389 ± 0.0156
Logits	0.9221 ± 0.0172	0.9368 ± 0.0174	0.9294 ± 0.0173
Features	**0.9427 ± 0.0018**	**0.9577 ± 0.0018**	**0.9501 ± 0.0018**
	Arrhythmia		
L_1	0.4588 ± 0.0248	0.4875 ± 0.0264	0.4727 ± 0.0256
L_2	0.4529 ± 0.0206	0.4813 ± 0.0219	0.4667 ± 0.0212
Logits	0.3706 ± 0.0834	0.3938 ± 0.0886	0.3818 ± 0.0859
Features	**0.5000 ± 0.0208**	**0.5313 ± 0.0221**	**0.5152 ± 0.0214**

We observe in Table VII that the adversarially learned features from the reconstruction discriminator are more suited for the anomaly detection task on tabular data.

Table VIII
DIFFERENT SCORE METHODS ON SVHN AND CIFAR-10 DATASETS

Anomaly Score	**AUROC**
SVHN	
L_1	0.5778 ± 0.0204
L_2	0.5826 ± 0.0201
Logits	0.5038 ± 0.0185
Features	0.5753 ± 0.0268
CIFAR-10	
L_1	0.6066 ± 0.1006
L_2	0.6012 ± 0.1088
Logits	0.5396 ± 0.0783
Features	0.6072 ± 0.1201

On image data, the features learned from the reconstruction discriminator lead to better detection of anomalies on the CIFAR-10 dataset, while performing comparably to the L_1 and L_2 variants on the SVHN dataset (Table VIII); full results are shown in the Appendix (Figures 9, 10).

VI. CONCLUSION

We present a GAN-based anomaly detection method ALAD that learns an encoder from the data space to the latent space during training, making it significantly more efficient at test time than the only published GAN method. In addition, we incorporate additional discriminators to improve the encoder, as well as spectral normalization which has been found to stabilize GAN training. Ablation analyses show that these modifications result in improved anomaly detection performance thus confirming their utility. Finally, we showed that ALAD is highly competitive with state-of-the-art anomaly detection methods on a range of tabular and image datasets and often outperforms them. While GANs can be difficult to train, the field is rapidly progressing and our method will directly benefit from any advances that accelerate or stabilize training. The effectiveness of ALAD positions GANs as a promising approach for anomaly detection on complex, high-dimensional data; applying ALAD to other data modalities such as speech and sensor data is an interesting direction for future research.

Acknowledgments

This research is supported by the Agency for Science, Technology and Research (A*STAR) under its SERC Strategic Funding (A1718g0045). The computational work for this article was partially performed on resources of the National Supercomputing Centre, Singapore (https://www.nscc.sg). We thank the reviewers for their feedback and suggestions, as well as Chunyuan Li and Hao Liu for helpful discussions related to the ALICE method.

References

[1] E. Schubert, A. Zimek, and H.-P. Kriegel, "Local outlier detection reconsidered: a generalized view on locality with applications to spatial, video, and network outlier detection," *Data Min. Knowl. Discov*, vol. 28, pp. 190-237, 2014.

[2] L. Martí, N. Sanchez-Pi, J. M. Molina, and A. C. B. Garcia, "Anomaly detection based on sensor data in petroleum industry applications," *Sensors*, vol. 15, no. 2, pp. 2774–2797, 2015.

[3] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comp. Sur.*, vol. 41, p. 15:1:15:58, 2009.

[4] A. Zimek, E. Schubert, and H.-P. Kriegel, "A survey on unsupervised outlier detection in high-dimensional numerical data," *Stat. Anal. Data Min.*, vol. 5, pp. 363-387, 2012.

[5] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Processing*, vol. 99, pp. 215 – 249, 2014.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 2672–2680.

[7] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *International Conference on Learning Representations, Workshop Track*, 2016.

[8] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 53–65, Jan 2018.

[9] H.-Y. L. Y. Tsao, "Generative adversarial network and its applications to speech signal and natural language processing," 2018. [Online]. Available: http://sigport.org/2863

[10] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," *International Conference on Information Processing in Medical Imaging*, p. pp. 146–157, 2017.

[11] V. Dumoulin, I. Belghazi, B. Poole, A. Lamb, M. Arjovsky, O. Mastropietro, and A. Courville, "Adversarially learned inference," *International Conference on Learning Representations*, 2017.

[12] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," *International Conference on Learning Representations*, 2017.

[13] C. Li, H. Liu, C. Chen, Y. Pu, L. Chen, R. Henao, and L. Carin, "Alice: Towards understanding adversarial learning for joint distribution matching," in *Advances in Neural Information Processing Systems 30*, 2017, pp. 5495–5503.

[14] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *International Conference on Learning Representations*, 2018.

[15] B. Schölkopf, R. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, "Support vector method for novelty detection," in *Proceedings of the 12th International Conference on Neural Information Processing Systems*, 1999, pp. 582–588.

[16] I. T. Jolliffe, *Principal component analysis*. Springer, New York, NY, 1986.

[17] S. Günter, N. N. Schraudolph, and S. V. N. Vishwanathan, "Fast iterative kernel principal component analysis," *Journal of Machine Learning Research*, pp. 1893–1918, 2007.

[18] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis," *Journal of the ACM*, 2009.

[19] M. Markou and S. Singh, "Novelty detection: a review—part 2: neural network based approaches," *Signal Processing*, vol. 83, no. 12, pp. 2499 – 2521, 2003.

[20] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 665-674, 2017.

[21] J. An and S. Cho, "Variational autoencoder based anomaly detection using reconstruction probability," *Technical Report*, 2015.

[22] S. Zhai, Y. Cheng, W. Lu, and Z. Zhang, "Deep structured energy based models for anomaly detection," *International Conference on Machine Learning*, pp. 1100-1109, 2016.

[23] Z. Bo, Q. Song, M. R. Min, W. C. C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," *International Conference on Learning Representations*, 2018.

[24] Y. Wu, Y. Burda, R. Salakhutdinov, and R. B. Grosse, "On the quantitative analysis of decoder-based generative models," *International Conference on Learning Representations*, 2017.

[25] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, "Generative visual manipulation on the natural image manifold," in *ECCV*, 2016.

[26] A. Creswell and A. A. Bharath, "Inverting the generator of A generative adversarial network," *NIPS Workshop on Adversarial Training*, 2016.

[27] Z. C. Lipton and S. Tripathi, "Precise recovery of latent vectors from generative adversarial networks," *CoRR*, vol. abs/1702.04782, 2017.

[28] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *Proceedings of Machine Learning Research*, vol. abs/1701.07875, 2017.

[29] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems 30*, 2017, pp. 5767–5777.

[30] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-Attention Generative Adversarial Networks," *ArXiv e-prints*, May 2018.

[31] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems 29*, 2016, pp. 2234–2242.

[32] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[33] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[34] A. Krizhevsky, "Learning multiple layers of features from tiny images," *Technical Report*, 2012.

[35] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 413–422.

[36] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

APPENDIX

A. CIFAR-10 and SVHN Experimental Details

Preprocessing: Pixels were scaled to be in range [-1,1].

1) DSEBM: For both CIFAR-10 and SVHN, we used the architecture suggested in [23]: one convolutional layer with kernel of size 3, strides of size 2, 64 filters and "same" padding, one max-pooling layer and one fully connected layer with 128 units.

2) AnoGAN: We took the official DCGAN architecture and hyper-parameters for these experiments. For the anomaly detection task, we took the same hyperparameters as the original paper. Similarly to ALAD, we used exponential moving average for inference with a decay of 0.999.

3) ALAD: The outputs of the underlined layer in the discriminator were used for the anomaly score. All convolutional layers have "same" padding unless specified otherwise.

Operation	Kernel	Strides	Filters Units	Non Linearity	Batch Norm.
$\boldsymbol{E(z)}$					
Conv2D	4×4	2×2	128	LReLU	✓
Conv2D	4×4	2×2	256	LReLU	✓
Conv2D	4×4	2×2	512	LReLU	✓
Conv2D*	4×4	1×1	100	None	×
$\boldsymbol{G(z)}$					
Trans. Conv2D*	4×4	2×2	512	ReLU	✓
Trans. Conv2D	4×4	2×2	256	ReLU	✓
Trans. Conv2D	4×4	2×2	128	ReLU	✓
Trans. Conv2D	4×4	2×2	3	Tanh	✓
$\boldsymbol{D_{xz}(x, z)}$					
Only on x					
Conv2D	4×4	2×2	128	LReLU	×
Conv2D	4×4	2×2	256	LReLU	✓
Conv2D	4×4	2×2	512	LReLU	✓
Only on z					
Conv2D†	4×4	2×2	512	LReLU	×
Conv2D†	4×4	2×2	512	LReLU	×
Concatenate outputs					
Conv2D†	1×1	1×1	1024	LReLU	×
Conv2D	1×1	1×1	1	LReLU	×
$\boldsymbol{D_{xx}(x, x')}$					
Concatenate x and x'					
Conv2D†	5×5	2×2	64	LReLU	×
$\underline{\text{Conv2D}^\dagger}$	5×5	2×2	128	LReLU	×
Dense			1	None	×
$\boldsymbol{D_{zz}(z, z')}$					
Concatenate z and z'					
Dense†			64	LReLU	×
Dense†			32	LReLU	×
Dense†			1	LReLU	×
Optimizer	Adam($\alpha = 2 * 10^{-4}$, $\beta_1 = 0.5$)				
Batch size	32				
Latent dimension	100				
Max Epochs	100				
Patience	10				
LReLU slope	0.2				
Weight &	Isotropic gaussian ($\mu = 0$, $\sigma = 0.01$)				
bias initialization	Constant(0)				

Table IX
CIFAR-10 AND SVHN ALAD ARCHITECTURE AND HYPERPARAMETERS
(† DROPOUT, * VALID PADDING)

B. KDD99 Experiment Details

Preprocessing: The dataset contains samples of 41 dimensions, where 34 of them are continuous and 7 are categorical. For categorical features, we further used one-hot representation to encode them; we obtained a total of 121 features after this encoding.

C. Arrhythmia Experiment Details

Preprocessing: The dataset contains samples of 274 dimensions. We applied our methods on raw data.

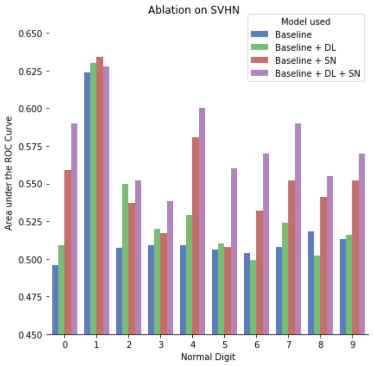

Figure 7. Ablation study on SVHN

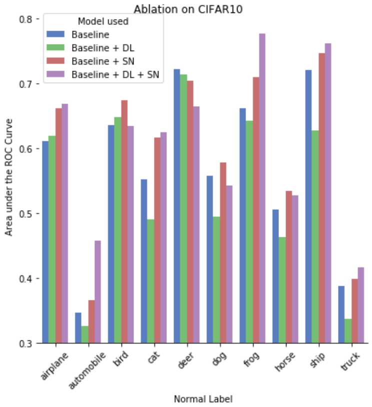

Figure 8. Ablation study on CIFAR-10

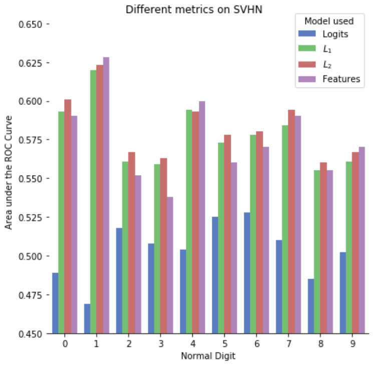

Figure 9. Performance of different anomaly scores on SVHN

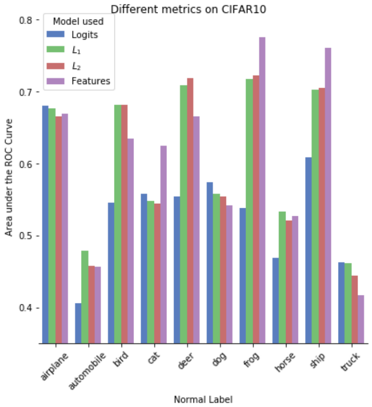

Figure 10. Performance of different anomaly scores on CIFAR-10

Operation	Units	Non Linearity	Dropout
$\boldsymbol{G(z)}$			
Dense	64	ReLU	0.0
Dense	128	ReLU	0.0
Dense	121	None	0.0
$\boldsymbol{D(x)}$			
Dense	256	LReLU	0.2
Dense	128	LReLU	0.2
Dense	128	LReLU	0.2
Dense	1	Sigmoid	0.0
Optimizer	Adam($\alpha = 10^{-5}$, $\beta_1 = 0.5$)		
Batch size	50		
Latent dimension	32		
Epochs	100		
LReLU slope	0.2		
Weight, bias initialization	Xavier Initializer, Constant(0)		

Table X
KDD99 GAN Architecture and hyperparameters

Operation	Units	Non Linearity	Batch Norm.	Dropout
$\boldsymbol{E(x)}$				
Dense	64	LReLU	×	0.0
Dense	1	None	×	0.0
$\boldsymbol{G(z)}$				
Dense	64	ReLU	×	0.0
Dense	128	ReLU	×	0.0
Dense	121	None	×	0.0
$\boldsymbol{D_{xz}(x, z)}$				
Only on x				
Dense	128	LReLU	✓	0.0
Only on z				
Dense	128	LReLU	×	0.5
Concatenate outputs				
Dense	128	LReLU	×	0.5
Dense	1	Sigmoid	×	0.0
$\boldsymbol{D_{xx}(x, x')}$				
Concatenate x and x'				
Dense	128	LReLU	×	0.2
Dense	1	Sigmoid	×	0.0
$\boldsymbol{D_{zz}(z, z')}$				
Concatenate z and z'				
Dense	32	LReLU	×	0.2
Dense	1	Sigmoid	×	0.0
Optimizer	Adam($\alpha = 10^{-5}$, $\beta_1 = 0.5$)			
Batch size	50			
Latent dimension	32			
Epochs	100			
LReLU slope	0.2			
Weight, bias init.	Xavier Initializer, Constant(0)			

Table XI
KDD99 ALAD Architecture and hyperparameters

Operation	Units	Non Linearity	Dropout
$\boldsymbol{G(z)}$			
Dense	128	ReLU	0.0
Dense	256	ReLU	0.0
Dense	274	None	0.0
$\boldsymbol{D(x)}$			
Dense	256	LReLU	0.2
Dense	128	LReLU	0.5
Dense	1	Sigmoid	0.0
Optimizer	Adam($\alpha = 10^{-5}$, $\beta_1 = 0.5$)		
Batch size	32		
Latent dimension	64		
Epochs	1000		
LReLU slope	0.2		
Weight, bias initialization	Xavier Initializer, Constant(0)		

Table XII
Arrhythmia GAN Architecture and hyperparameters

Operation	Units	Non Linearity	Batch Norm.	Dropout
$\boldsymbol{E(x)}$				
Dense	256	LReLU	×	0.0
Dense	128	LReLU	×	0.0
Dense	64	None	×	0.0
$\boldsymbol{G(z)}$				
Dense	128	ReLU	×	0.0
Dense	256	ReLU	×	0.0
Dense	274	None	×	0.0
$\boldsymbol{D_{xz}(x, z)}$				
Only x				
Dense	128	LReLU	√	0.0
Only z				
Dense	128	LReLU	×	0.5
Concatenate outputs				
Dense	256	LReLU	×	0.5
Dense	1	Sigmoid	×	0.0
$\boldsymbol{D_{xx}(x, x')}$				
Concatenate x and x'				
Dense	256	LReLU	×	0.2
Dense	128	LReLU	×	0.2
Dense	1	Sigmoid	×	0.0
$\boldsymbol{D_{zz}(z, z')}$				
Concatenate z and z'				
Dense	64	LReLU	×	0.2
Dense	32	LReLU	×	0.2
Dense	1	Sigmoid	×	0.0
Optimizer	Adam($\alpha = 10^{-5}, \beta_1 = 0.5$)			
Batch size	32			
Latent dimension	64			
Epochs	1000			
LReLU slope	0.2			
Weight, bias init.	Xavier Initializer, Constant(0)			

Table XIII

Arrhythmia ALAD Architecture and hyperparameters