Fence GAN: Towards Better Anomaly Detection

Phuc Cuong Ngo*
Hwa Chong Institution (College Section)
Singapore
141958r@student.hci.edu.sg

Amadeus Aristo Winarto*
Hwa Chong Institution (College Section)
Singapore
141911b@student.hci.edu.sg

Connie Khor Li Kou
Bioinformatics Institute,
*Agency for Science, Technology and Research (A*STAR)*
School of Computing, National University of Singapore
Singapore
koukl@bii.a-star.edu.sg

Sojeong Park
Bioinformatics Institute
*Agency for Science, Technology and Research (A*STAR)*
Singapore
Department of Physics, Chosun University
South Korea
parksj@bii.a-star.edu.sg

Farhan Akram
Bioinformatics Institute,
*Agency for Science, Technology and Research (A*STAR)*
Singapore
farhana@bii.a-star.edu.sg

Hwee Kuan Lee
Bioinformatics Institute,
*Agency for Science, Technology and Research (A*STAR)*
School of Computing, National University of Singapore
Singapore Eye Research Institute (SERI)
Image and Pervasive Access Laboratory (IPAL)
Singapore
leehk@bii.a-star.edu.sg

***Abstract*—Anomaly detection is a classical problem where the aim is to detect anomalous data that do not belong to the normal data distribution. Current state-of-the-art methods for anomaly detection on complex high-dimensional data are based on the generative adversarial network (GAN). However, the traditional GAN loss is not directly aligned with the anomaly detection objective: it encourages the distribution of the generated samples to overlap with the real data and so the resulting discriminator is ineffective as an anomaly detector. In this paper, we propose modifications to the GAN loss such that the generated samples lie at the boundaries of the real data distribution. With our modified GAN loss, our anomaly detection method, called Fence GAN (FGAN), directly uses the discriminator score as an anomaly threshold. Our experimental results on the MNIST, CIFAR10 and KDD99 datasets show that FGAN yields the best anomaly classification accuracy compared to state-of-the-art methods.**

***Index Terms*—Generative Adversarial Network, Anomaly Detection**

I. Introduction

Anomaly detection is a well-known problem in artificial intelligence where one aims to identify anomalous instances that do not belong to the normal data distribution [5, 14]. It is used in a wide range of applications such as network intrusion [10], credit card fraud [29], crowd surveillance [20, 23], healthcare [25] and many more. Traditional classifiers trained in a supervised setting do not work well in anomaly detection since the anomalous data is usually unavailable or very few. Hence, anomaly detectors are usually trained in an unsupervised setting where the distribution of the normal data is learned and instances that are unlikely to be under this distribution are identified as anomalous.

For complex high-dimensional datasets such as images, recent methods based on generative adversarial networks (GANs) have shown state-of-the-art anomaly detection performance by exploiting GANs ability to model high-dimensional data distributions. However, we identify a shortcoming of current GAN-based anomaly detection methods: the usual GAN objective encourages the distribution of generated samples to overlap with the real data, and this is not directly aligned with the anomaly detection objective. The resulting discriminator is not effective at differentiating between real and anomalous data. In this paper, we propose modifications to the GAN objective such that the generated samples lie at the boundaries of the real data distribution instead of overlapping it. Our method, which we call Fence GAN (FGAN), trains in the usual adversarial manner with the modified objective and we show that the resulting discriminator can be used as an anomaly detector. We conducted experiments on MNIST, CIFAR10 and KDD99 datasets and show that FGAN outperforms state-of-the-art anomaly detection methods [1, 25, 31, 32].

II. Related Work

Traditional methods for anomaly detection include one-class SVM [26], nearest neighbor [9], clustering [28], kernel density estimation [21] and hidden markov models [13]. However, such methods are not suitable for high-dimensional image data. Recent developments in deep learning have led to significant progress in supervised learning tasks on complex image

*Equal contribution

2375-0197/19/$31.00 ©2019 IEEE
DOI 10.1109/ICTAI.2019.00028

datasets [15, 24]. For anomaly detection, deep learning based methods include deep belief networks [8], variational autoencoders [2, 30] and adversarial autoencoders [4, 18, 19, 34].

Among the deep learning methods, generative adversarial networks (GANs) [11, 22] have been the subject of extensive research as they show state-of-the-art performance in modeling complex high-dimensional image distributions. Similarly, GANs have been used for anomaly detection. In AnoGAN by Schlegl et al. [25], the authors propose an anomaly detector where the GAN is first trained on the normal images, and for a test image, the latent space is iteratively searched to find the latent vector that best reconstructs the test image. The anomaly score is a combination of the reconstruction loss and the loss between the intermediate discriminator feature of the test image and the reconstructed image. A similar framework is used in ADGAN [7] where the anomaly score is based only on reconstruction loss, the search in latent space is repeated with multiple seeds and both the latent vector and generator are optimized. More recent methods, named Efficient GAN-based Anomaly Detection (EGBAD) and its improved version Adversarially Learned Anomaly Detection (ALAD) by Zenati et al. [31, 32], make use of the Bidirectional GAN model that is able to map from the image to latent space without iterative search, resulting in superior anomaly detection performance and faster test times. Finally, in the GANomaly framework [1], the generator consists of encoder-decoder-encoder subnetworks and the anomaly score is based on a combination of encoding, reconstruction and feature matching losses. GANomaly has shown superior performance compared to AnoGAN and ALAD on several image datasets such as MNIST and CIFAR10.

Except for GANomaly, the GAN-based anomaly detection methods above train GAN with the usual minimax loss function where the generator aims to generate samples that overlap with the data distribution. The discriminator probability score trained under the usual GAN loss function has found to be ineffective [7], and we hypothesize that this is because the discriminator is not explicitly trained to fence the boundaries of the data distribution. Contrary to these methods, our proposed FGAN aims to learn the boundaries of the normal data distribution. We achieve this by modifying the generator's objective to generate data lying on the boundaries of the normal data distribution instead of overlapping with the data distribution. At test time, the anomaly score is simply the discriminator score given to the input data. Our alternative generator objective is similar to the one in Dai et al. [6], where they show that for the discriminator to be a good classifier, the generator has to produce complement samples instead of matching with the true data distribution. With the modified GAN loss, FGAN does not need to rely on reconstruction loss from the generator and does not require modifications to the basic GAN architecture unlike ALAD and GANomaly.

III. Method

A. Original GAN loss function

In the original generative adversarial network by Goodfellow et al. [12], for a set $\mathcal{X}$ of N number of data points $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N\}$ with $\boldsymbol{x}_i$ in a Euclidean data space $\mathbb{R}^d$, $d \in \mathbb{Z}^+$, $i = 1, 2, .., N$, which is sampled from a data distribution $p_{data} : \mathbb{R}^d \to \mathbb{R}^+$, we seek to map points from a prior noise distribution $p_{noise} : \mathbb{R}^k \to \mathbb{R}^+$, $k \in \mathbb{Z}^+$ to p_{data}. For example, if each data point represents an image, then d would be the number of pixels in the image. The dimension k is set arbitrarily.

The mapping from p_{noise} to p_{data} is done by first using a differentiable function, represented by a generator network G_θ with θ being its weights and biases, to map p_{noise} to the generated distribution $p_g : \mathbb{R}^d \to \mathbb{R}^+$ from the output $G_\theta(\boldsymbol{z}), G_\theta(\boldsymbol{z}) \in \mathbb{R}^d$ of G_θ and $\boldsymbol{z}$ is drawn from p_{noise}. In addition, we also have a discriminator network D_ϕ with ϕ being its weights and biases, which outputs a real value $D_\phi(\boldsymbol{x}) \in [0, 1]$ that represents the probability of $\boldsymbol{x}$ being drawn from p_{data} rather than from p_g. D_ϕ and G_θ engage in a two-player minimax game, with D_ϕ and G_θ being alternatingly trained to minimize their respective loss functions as follows:

$$\mathcal{L}_{G_\theta}^{GAN}(G_\theta, D_\phi, \mathcal{Z}) = \frac{1}{N}\sum_{i=1}^{N}\left[\log(1 - D_\phi(G_\theta(\boldsymbol{z}_i)))\right] \quad (1)$$

$$\mathcal{L}_{D_\phi}{}^{GAN}(G_\theta, D_\phi, \mathcal{X}, \mathcal{Z}) = \frac{1}{N}\sum_{i=1}^{N}\left[-\log(D_\phi(\boldsymbol{x}_i)) - \log(1 - D_\phi(G_\theta(\boldsymbol{z}_i)))\right] \quad (2)$$

where $\mathcal{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_N\}$ is sampled from p_{noise}. $\mathcal{L}_{G_\theta}^{GAN}$ is the loss function of G_θ and $\mathcal{L}_{D_\phi}^{GAN}$ is the loss function of D_ϕ.

In this way, D_ϕ is trained to differentiate whether $\boldsymbol{x}$ is drawn from p_{data} or from p_g. Meanwhile, G_θ is trained to map p_{noise} to p_g so as to maximise the score of its generated points as given by the discriminator, that is, $D_\phi(G_\theta(\boldsymbol{z}))$.

The training of GAN is completed if the distribution p_g is indistinguishable from p_{data}. When this occurs, p_{data} is estimated by p_g. Therefore, the mapping of points from p_{noise} to p_g, which is represented by G_θ, is also the mapping of points from p_{noise} to p_{data}.

B. Modified loss functions

For the anomaly detection task, we propose FGAN which has a different objective from the original GAN. Whereas the original GAN aims to generate $p_g = p_{data}$, that is, to generate points at regions of high data density, our objective is to generate points at regions of low data density and around the boundaries of $\mathcal{X}$, which we denote as $\delta\mathcal{X}$. This will enable our discriminator, at the end of training, to draw boundaries "tightly" around $\mathcal{X}$. Such a discriminator can then be used as a one-class classifier or an anomaly detector.

Learning $\delta\mathcal{X}$ directly is known to be an extremely difficult problem in high dimensions [27]. Thus, we use the discriminator score to define the domain of $\delta\mathcal{X}$ and then estimate $\delta\mathcal{X}$ using the generator in FGAN. The generated points $G_\theta(\boldsymbol{z})$ then must enclose the real data points tightly as shown in Figure 1(A). In order to achieve our objective, we propose a series of modifications to the loss functions for the generator and discriminator: encirclement and dispersion losses for generator, and weighted discriminator loss for discriminator.

1) Generator encirclement loss: In our proposed FGAN, we want the generator to generate points $G_\theta(\boldsymbol{z})$ that lie in $\delta\mathcal{X}$, at the boundaries of $\mathcal{X}$. To achieve this, we modify the generator loss such that the generator aims for the generated samples to have discriminator scores between $(0, 1)$ instead of the maximum score of 1 as in the usual GAN loss. To reflect this, the loss function of the generator in FGAN is:

$$EL\big(G_\theta, D_\phi, \mathcal{Z}\big) = \frac{1}{N}\sum_{i=1}^{N}\left[\log(|\alpha - D_\phi(G_\theta(\boldsymbol{z}_i))|)\right] \quad (3)$$

where $\alpha \in (0, 1)$ is used for the generator to generate points on $\delta\mathcal{X}$. The rationale for Eq. (3) is that points generated inside of $\mathcal{X}$ and points generated far from $\mathcal{X}$ will be penalized. When the points are generated at the α-level set of the discriminator score, the generator will achieve minimal loss. This level set should ideally tightly enclose the real data points. In our experiments, we tune the value of α as a hyperparameter.

2) Generator dispersion loss: Based on the encirclement loss alone, however, there is no guarantee that the generated points will cover the entirety of $\delta\mathcal{X}$, it may only cover a small part of it. We note that this is similar to the mode collapse problem in GAN. The dispersion loss, which maximizes distance of the generated data points from their centre of mass $\boldsymbol{\mu}$, $\boldsymbol{\mu} \in \mathbb{R}^d$, is thus introduced to encourage the generated points to cover all boundaries.

$$\boldsymbol{\mu} = (\mu_1, \mu_2, ..., \mu_d)$$
$$\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N} G_\theta(\boldsymbol{z}_i)$$

The dispersion loss is thus:

$$DL(G_\theta, \mathcal{Z}) = \frac{1}{\frac{1}{N}\sum_{i=1}^{N}(\|G_\theta(\boldsymbol{z}_i) - \boldsymbol{\mu}\|_2)} \quad (4)$$

We use L_2 distance because in our experiments we found that it works better compared to L_1 and L_∞ distances. The loss function of the generator in FGAN to be minimized is the weighted sum of the encirclement loss and the dispersion loss:

$$\begin{aligned}\mathcal{L}^{FGAN}_{generator}\big(G_\theta, D_\phi, \mathcal{Z}\big) &= EL + \beta \times DL \\ &= \frac{1}{N}\sum_{i=1}^{N}\left[\log\Big[|\alpha - D_\phi(G_\theta(\boldsymbol{z}_i))|\Big]\right] \\ &\quad + \beta \times \frac{1}{\frac{1}{N}\sum_{i=1}^{N}(\|G_\theta(\boldsymbol{z}_i) - \boldsymbol{\mu}\|_2)}\end{aligned} \quad (5)$$

where β is the dispersion hyperparameter with $\beta \in \mathbb{R}^+$.

Algorithm 1 Stochastic gradient descent training of FGAN.

for number of training iterations **do**

• Sample noise samples $\mathcal{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_N\}$ from prior p_{noise}

• Update the generator's parameters:

$$\theta \leftarrow \theta - \eta_g \nabla_\theta \mathcal{L}^{FGAN}_{generator}\big(G_\theta, D_\phi, \mathcal{Z}\big)$$

• Resample noise samples $\mathcal{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_N\}$ from prior p_{noise}

• Sample data samples $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N\}$ from real data distribution p_{data}

• Update the discriminator's parameters:

$$\phi \leftarrow \phi - \eta_d \nabla_\phi \mathcal{L}^{FGAN}_{discriminator}\big(G_\theta, D_\phi, \mathcal{Z}, \mathcal{X}\big)$$

end for

The generator and discriminator learning rates η_g and η_d are hyperparameters.

3) Discriminator weighted discriminator loss: As the generator becomes better in approximating $\delta\mathcal{X}$, the discriminator faces a trade-off: to classify real data correctly or classify generated data correctly. If the discriminator focuses more on classifying generated data correctly, then the discriminator will start to classify real data as generated data. Thus, the loss function of the discriminator should be modified to prioritise classifying real data correctly:

$$\begin{aligned}&\mathcal{L}^{FGAN}_{discriminator}\big(G_\theta, D_\phi, \mathcal{X}, \mathcal{Z}\big) = \\ &\frac{1}{N}\sum_{i=1}^{N}\left[-\log\big(D_\phi(\boldsymbol{x}_i)\big) - \gamma\log\Big(1 - D_\phi(G_\theta(\boldsymbol{z}_i))\Big)\right]\end{aligned} \quad (6)$$

where γ is the anomaly hyperparameter with $\gamma \in (0, 1]$. When γ is less than 1, the discriminator will focus more on classifying the real data points correctly, thus its decision boundary is less likely to bend into the domain of $\mathcal{X}$, allowing the generator to better estimate $\delta\mathcal{X}$. We empirically tune γ for each dataset.

C. FGAN

FGAN is composed of a generator and a discriminator being trained one after another like a typical GAN. The number of steps to train G_θ and D_ϕ for each iteration is a hyperparameter to be tuned. However for simplicity, we train both networks once in each iteration. The algorithm to train FGAN is described in Algorithm 1.

IV. Experiments

In our experiments, we tested FGAN on a synthetic 2D dataset to study the training process in FGAN. Next, we tested FGAN for anomaly detection on three datasets: MNIST, CIFAR10 and KDD99, comparing the performance to state-of-the-art anomaly detection methods. Our code is available online at https://github.com/phuccuongngo99/Fence_GAN.

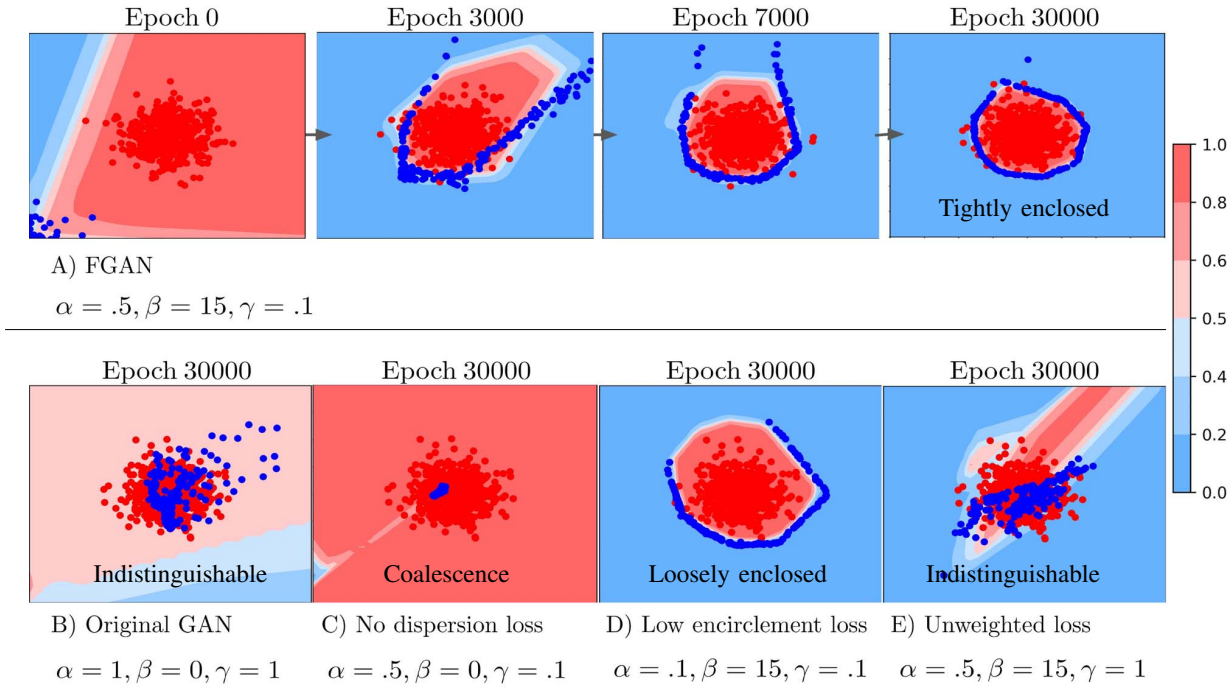

Fig. 1. FGAN on 2D normal distribution with red data points and blue generated points. The color of the shaded background represents the discriminator score. We trained 5 models with different hyperparameters for 30000 epochs. (A) shows snapshots of training process for a FGAN with optimal hyperparameters, giving a good discriminator in the end of the training. (B)-(E) are examples of hyperparameters that lead to suboptimal performance of FGAN.

A. 2D synthetic dataset

We illustrate the effect of FGAN using a 2D synthetic dataset where the data is sampled from a unimodal normal distribution, as shown in Figure 1. The red points are the real data while the blue points are the generated points. The color of the shaded background represents the discriminator score, where the score increases from blue to red. We trained 5 different FGAN models with different hyperparameters over 30,000 epochs. In (A), we show snapshots of training process at 4 epochs for an FGAN trained with optimal hyperparameters, yielding a good discriminator at the end of the training. In contrast, (B) shows the result of original GAN by Goodfellow et al. [12], where the real data points and generated data points are indistinguishable and the discriminator decision boundary does not surround the real data. Because of the modified loss functions in FGAN, over the training process in (A), the generated samples approach the boundaries of the real data without entering the data distribution. As a result of this fencing behavior, the discriminator gives low scores to all data points outside of the real data distribution and hence is effective at anomaly detection. However, in the original GAN in (B), the loss function encourages overlap of the generated samples with the real data and the discriminator gives similar scores to data points within and out of the real data distribution.

The other examples show hyperparameters that lead to suboptimal performance. (C) illustrates an example of generated points coalescing in one small region and the discriminator classifies most of the data space as positive instances. (D) shows an example of loosely enclosed generated points where the discriminator decision boundary is away from the real data points. (E) shows another example of indistinguishable real data and generated data distributions with bad discriminator decision boundaries. This experiment on the 2D dataset shows that under the optimal hyperparameters, the encirclement loss, dispersion loss and weighted discriminator loss in FGAN give rise to the desired result of the generated samples forming tight boundaries around the dataset.

B. MNIST

To show the effectiveness of our proposed idea, we run FGAN for anomaly detection on the MNIST dataset [17]. In each case, we consider data points from a class as ‘anomalous’ (positive class) and data points from the other 9 classes as ‘normal’ (negative class). We then split the entire MNIST dataset which consists of 70000 images from 10 classes into 2 sets as follows: Training set consists of 80% of all data points in the ‘normal’ class. Testing set consists of the rest 20% of data in the ‘normal’ class and all data in the ‘anomalous’ class. We then evaluate our model as a binary classifier for normal and anomalous data. Our performance is measured by the Area

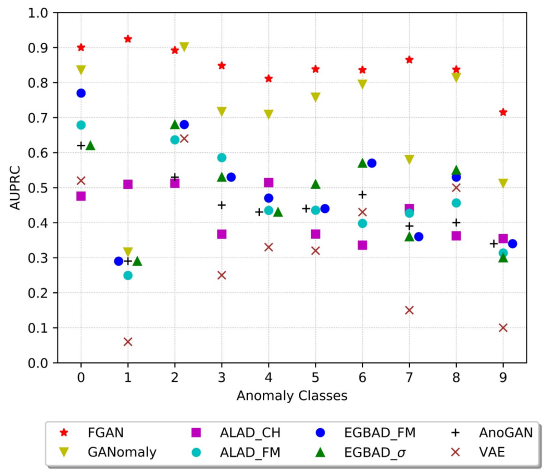

Fig. 2. Detection accuracies on the MNIST dataset.

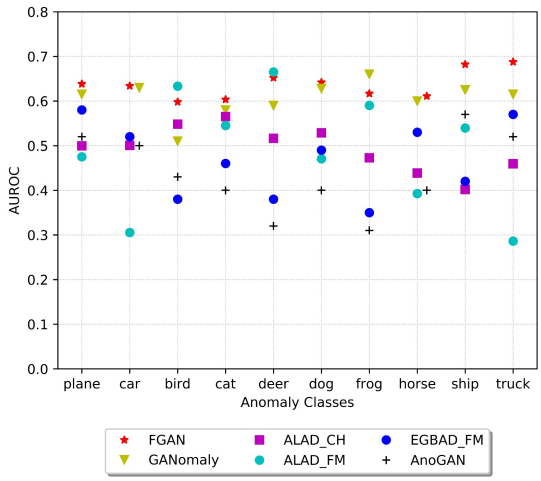

Fig. 3. Detection accuracies on the CIFAR10 dataset.

Under Precision and Recall Curve (AUPRC). The architecture as well as hyperparameters to train FGAN are presented in Table II.

We train FGAN for 100 epochs with a training batch size of 100 and obtain the average AUPRC across 3 independent, identical experiments with different random initializations for each anomalous class. Mean AUPRC in comparison to other benchmark methods are shown in Figure 2. The other benchmark methods are trained using the same setup for splitting of the train and test sets. We reimplement ALAD [32] and GANomaly [1] using the hyperparameters used by the authors to obtain the AUPRC while the results for EGBAD, AnoGAN and VAE were taken from Zenati et al. [31]. ALAD_FM and EGBAD_FM use feature matching to score anomalies, while ALAD_CH and EGBAD_σ use the cross-entropy loss of the discriminator to score anomalies [31, 32]. As seen from the AUPRC figures, FGAN has the highest accuracy for all but one digit class. Interestingly, for digit classes where the other methods perform badly (eg. digits 1, 7, 9), FGAN's detection accuracy remains high. This shows FGAN's superior performance in anomaly detection for the MNIST dataset.

C. CIFAR10

Next, we run the on the CIFAR10 dataset [16]. Similar to the MNIST experiment, we consider images from a particular class as 'anomalous' and images from the other 9 classes as 'normal'. We split the entire CIFAR10 dataset of 60000 images into a train set consisting of 80% of 'normal' class images and a test set consisting of the remaining 20% of 'normal' class images and all of the images in the 'anomalous' class. We train FGAN for 150 epochs with batch size of 128. The performance is measured by the Area Under Receiver Operating Characteristics (AUROC) curve, averaged over 3 independent, identical experiments with different random initializations. The network architecture and hyperparameters to train FGAN are in Table III.

The anomaly detection results are shown in Figure 3. The results for GANomaly, EGBAD_FM and AnoGAN were taken from Akcay et al. [1]. We reimplement ALAD_CH and ALAD_FM by Zenati et al. [32] using the hyperparameters used by the authors due to slight differences in the dataset splitting and then average the results across three consecutive seeds. Across all classes, FGAN achieves an AUROC of at least 60%. Furthermore, FGAN performs similarly well across all classes whereas ALAD_FM and GANomaly show larger range of AUROC across all classes, showing FGAN's strength at anomaly detection.

Figure 4 shows the average discriminator score for the anomalous 'ship' class, the other 9 classes and the generated images. All scores are taken from the test set. Interestingly the 'airplane' class has a lower score than the anomalous 'ship' class, and this may indicate that these two classes are semantically similar and challenging to differentiate compared to other classes.

In Figure 5, we analyze the distribution of discriminator scores for the normal test images, anomalous images and generated images where the anomalous class is 'ship'. Each histogram is normalized such that the sum of area of the bins equals one. The normal test images scores are skewed towards the high score of 1.0 which is expected from the FGAN loss function. The anomalous images scores show a bimodal distribution with modes at 0.3 and 1.0, which means some anomalous images are challenging to detect. This explains the relatively good anomaly detection results in Figure 3. The distribution of scores for generated images is spread across the entire range which means the generated samples do not converge at a score of $\alpha = 0.5$, although this has not significantly impacted anomaly detection performance.

D. KDD99

In order to further validate the merits of our approach, we test FGAN on the KDDCUP99 10 percent dataset [3]. We

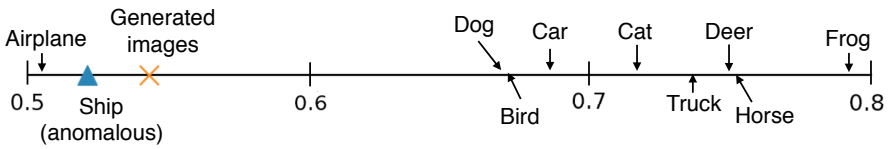

Fig. 4. Average discriminator scores for the anomalous 'ship' class, the other 9 classes and the generated images. All scores were taken from the test set.

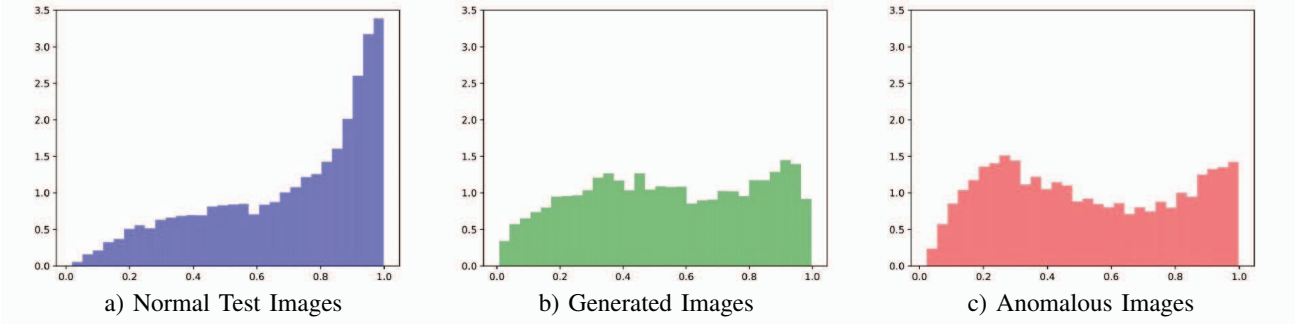

Fig. 5. CIFAR10 anomalous class: ship. Histogram of discriminator scores for (a) normal test images, (b) generated images and (c) anomalous images. The normal test images have a distribution that is skewed towards higher scores, which is expected from the FGAN loss function.

follow the experimental setup of Zong et al. [35] and Zenati et al. [32] in our experiments. In the KDD99 dataset, the data is stratified into the 'non-attack' class and other classes with various attacks. We lump all the other classes with various attacks as one class and call it the 'attack' class. We then train FGAN on the 'attack' class only because the proportion of data belonging to the 'attack' class is much larger than the proportion of data belonging to the 'non-attack' class. The objective is to detect the 'non-attack' instances.

We split the KDD99 dataset as follows: Training set consists of 50% of all data points in the 'attack' class. Testing set consists of the remaining 50% of data in the 'attack' class and 50% of data in the 'non-attack' class. The architecture and hyperparameters to train FGAN for KDD99 dataset is shown in Table IV. We train with a batch size of 256 for both discriminator and generator for 50 epochs. The precision, recall and F1 scores are calculated with 'non-attack' class being the positive class and 'attack' class being the negative class, as in Zenati et al. [31, 32], and are averaged over 10 consecutive seeds, as shown in Table I. Values for OC-SVM, DSEBM, DAGMM were obtained from Zhai et al. [33], Zong et al. [35]. Values for AnoGAN and ALAD were obtained from Zenati et al. [32]. FGAN has the best anomaly detection accuracies compared to the other benchmark methods.

TABLE I
PERFORMANCE OF FGAN AND BENCHMARK METHODS THE KDD99 DATASET.

Model	Precision	Recall	F1
OC-SVM	0.7457	0.8523	0.7954
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
FGAN	**0.9546**	**0.9697**	**0.9553**

V. DISCUSSION

State-of-the-art anomaly detection methods for complex high-dimensional data are based on generative adversarial networks. However, in this paper, we identify that the usual GAN loss objective is not directly aligned with the anomaly detection objective: the loss encourages the distribution of generated samples to overlap with real data. Hence, the resulting discriminator has been found to be ineffective for anomaly detection. We propose simple modifications to the GAN loss such that the generated samples lie at the boundaries of the real data distribution. Our method, called FGAN, uses the discriminator score as anomaly score. With the modified GAN loss, FGAN does not need to rely on reconstruction loss from the generator and does not require modifications to the basic GAN architecture unlike ALAD and GANomaly. On the MNIST, CIFAR10 and KDD99 datasets, FGAN outperforms existing methods in anomaly detection. We have shown that with simple modifications to the GAN loss, the basic GAN architecture and training scheme can produce an effective anomaly detector for complex high-dimensional data.

REFERENCES

[1] Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. *arXiv preprint arXiv:1805.06725*, 2018.

[2] Jinwon An and Sungzoon Cho. Variational autoencoder

based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2:1–18, 2015.

[3] Stephen D Bay, Dennis F Kibler, Michael J Pazzani, and Padhraic Smyth. The uci kdd archive of large data sets for data mining research and experimentation. *SIGKDD Explorations*, 2:81, 2000.

[4] Laura Beggel, Michael Pfeiffer, and Bernd Bischl. Robust anomaly detection in images using adversarial autoencoders. *arXiv preprint arXiv:1901.06355*, 2019.

[5] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[6] Zihang Dai, Zhilin Yang, Fan Yang, William W Cohen, and Ruslan R Salakhutdinov. Good semi-supervised learning that requires a bad gan. In *Advances in Neural Information Processing Systems*, pages 6510–6520, 2017.

[7] Lucas Deecke, Robert Vandermeulen, Lukas Ruff, Stephan Mandt, and Marius Kloft. Anomaly detection with generative adversarial networks. 2018.

[8] Sarah M Erfani, Sutharshan Rajasegarar, Shanika Karunasekera, and Christopher Leckie. High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning. *Pattern Recognition*, 58:121–134, 2016.

[9] Eleazar Eskin, Andrew Arnold, Michael Prerau, Leonid Portnoy, and Sal Stolfo. A geometric framework for unsupervised anomaly detection. In *Applications of data mining in computer security*, pages 77–101. Springer, 2002.

[10] Pedro Garcia-Teodoro, Jesus Diaz-Verdejo, Gabriel Maciá-Fernández, and Enrique Vázquez. Anomaly-based network intrusion detection: Techniques, systems and challenges. *computers & security*, 28(1-2):18–28, 2009.

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

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

[13] Nico Görnitz, Mikio Braun, and Marius Kloft. Hidden markov anomaly detection. In *International Conference on Machine Learning*, pages 1833–1842, 2015.

[14] Douglas M Hawkins. *Identification of outliers*, volume 11. Springer, 1980.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[16] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[17] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL `http://yann.lecun.com/exdb/mnist/`.

[18] Valentin Leveau and Alexis Joly. Adversarial autoencoders for novelty detection. 2017.

[19] Swee Kiat Lim, Yi Loo, Ngoc-Trung Tran, Ngai-Man Cheung, Gemma Roig, and Yuval Elovici. Doping: Generative data augmentation for unsupervised anomaly detection with gan. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 1122–1127. IEEE, 2018.

[20] Vijay Mahadevan, Weixin Li, Viral Bhalodia, and Nuno Vasconcelos. Anomaly detection in crowded scenes. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1975–1981. IEEE, 2010.

[21] Miguel Nicolau, James McDermott, et al. One-class classification for anomaly detection with kernel density estimation and genetic programming. In *European Conference on Genetic Programming*, pages 3–18. Springer, 2016.

[22] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[23] Mahdyar Ravanbakhsh, Enver Sangineto, Moin Nabi, and Nicu Sebe. Training adversarial discriminators for cross-channel abnormal event detection in crowds. *arXiv preprint arXiv:1706.07680*, 2017.

[24] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[25] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International Conference on Information Processing in Medical Imaging*, pages 146–157. Springer, 2017.

[26] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

[27] Raimund Seidel. Constructing higher-dimensional convex hulls at logarithmic cost per face. pages 404–413, 01 1986. doi: 10.1145/12130.12172.

[28] Rasheda Smith, Alan Bivens, Mark Embrechts, Chandrika Palagiri, and Boleslaw Szymanski. Clustering approaches for anomaly based intrusion detection. *Proceedings of intelligent engineering systems through artificial neural networks*, pages 579–584, 2002.

[29] Abhinav Srivastava, Amlan Kundu, Shamik Sural, and Arun Majumdar. Credit card fraud detection using hidden markov model. *IEEE Transactions on dependable and secure computing*, 5(1):37–48, 2008.

[30] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. Unsupervised anomaly detection

via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 187–196. International World Wide Web Conferences Steering Committee, 2018.

[31] Houssam Zenati, Chuan Sheng Foo, Bruno Lecouat, Gaurav Manek, and Vijay Ramaseshan Chandrasekhar. Efficient GAN-Based Anomaly Detection. *arXiv e-prints*, art. arXiv:1802.06222, Feb 2018.

[32] Houssam Zenati, Manon Romain, Chuan Sheng Foo, Bruno Lecouat, and Vijay Ramaseshan Chandrasekhar. Adversarially Learned Anomaly Detection. *arXiv e-prints*, art. arXiv:1812.02288, Dec 2018.

[33] Shuangfei Zhai, Yu Cheng, Weining Lu, and Zhongfei Zhang. Deep structured energy based models for anomaly detection. *CoRR*, abs/1605.07717, 2016. URL http://arxiv.org/abs/1605.07717.

[34] Shuangfei Zhai, Yu Cheng, Weining Lu, and Zhongfei Zhang. Deep structured energy based models for anomaly detection. *arXiv preprint arXiv:1605.07717*, 2016.

[35] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BJJLHbb0-

APPENDIX

TABLE II
ARCHITECTURE AND HYPERPARAMETERS OF FGAN ON MNIST. (BN: BATCH NORM, FM: FEATURE MAPS, LR: LEARNING RATE)

Operation	Kernel	FMs/Units	BN?	Activation
Generator				
Dense		1024	✓	ReLU
Dense		7×7×128	✓	ReLU
Transposed Convolution	4×4	64	✓	ReLU
Transposed Convolution	4×4	1	×	Tanh
Latent Dimension	200			
Encirclement α	0.1			
Dispersion β	30			
Optimizer	Adam(lr=2e-5, decay=1e-4)			
Discriminator				
Convolution	4×4	64	×	Leaky ReLU
Convolution	4×4	64	×	Leaky ReLU
Dense		1024	×	Leaky ReLU
Dense		1	×	Sigmoid
Leaky ReLU slope	0.1			
Optimizer	Adam(lr=1e-5, decay=1e-4)			
Anomaly γ	0.1			
Epochs	100			
Batchsize	200			

TABLE III
ARCHITECTURE AND HYPERPARAMETERS OF FGAN ON CIFAR-10. (BN: BATCH NORM, FM: FEATURE MAPS, LR: LEARNING RATE)

Operation	Kernel	FMs/Units	BN?	Activation
Generator				
Dense		2×2×256	✓	Leaky ReLU
Transposed Convolution	5×5	128	✓	Leaky ReLU
Transposed Convolution	5×5	64	✓	Leaky ReLU
Transposed Convolution	5×5	32	✓	Leaky ReLU
Transposed Convolution	5×5	3	×	Tanh
Latent Dimension	256			
Leaky ReLU Slope	0.2			
Encirclement α	0.5			
Dispersion β	10			
Optimizer	Adam(lr=1e-3, beta_1 = 0.5, beta_2 = 0.999)			
Decay	decay=1e-5			
Discriminator				
Convolution	5×5	32	✓	Leaky ReLU
Convolution	5×5	64	✓	Leaky ReLU
Convolution	5×5	128	✓	Leaky ReLU
Convolution	5×5	256	×	Leaky ReLU
Dropout		0.2	×	
Dense		1	×	Sigmoid
Leaky ReLU Slope	0.2			
Weight Decay	0.5			
Optimizer	Adam(lr=1e-4, beta_1 = 0.5, beta_2 = 0.999)			
Decay	decay=1e-5			
Anomaly γ	0.5			
Epochs	150			
Batch Size	128			

TABLE IV
ARCHITECTURE AND HYPERPARAMETERS FOR FGAN ON KDD99. (BN: BATCH NORM, FM: FEATURE MAPS, LR: LEARNING RATE)

Operation	Units	Activation	Dropout	L2 Reg
Generator			0	0
Dense	64	ReLU	0.2	0
Dense	128	ReLU	0.2	0
Dense (output)	121	Linear	0	0
Latent Dimension	32			
Encirclement α	0.5			
Dispersion β	30			
Optimizer	Adam(lr = 1e-4, decay = 1e-3)			
Discriminator			0	
Dense	256	Leaky ReLU	0	0
Dense	128	Leaky ReLU	0	0
Dense	128	Leaky ReLU	0	0
Dense (output)	1	Sigmoid	0	
Leaky ReLU Slope	0.1			
Optimizer	SGD(lr = 8e-6, decay = 1e-3)			
Anomaly γ	0.5			
Epochs	50			
Batch Size	256			