Neurocomputing 537 (2023) 296–308

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A step-by-step training method for multi generator GANs with application to anomaly detection and cybersecurity

Mohammad Adiban[a,b], Sabato Marco Siniscalchi[a], Giampiero Salvi[a,c]

[a] Department of Electronic Systems, NTNU, Trondheim, Norway
[b] Department of Human Centred Computing, Monash University, Melbourne, Australia
[c] KTH Royal Institute of Technology, School of Electrical Engineering and Computer Science, Stockholm, Sweden

ARTICLE INFO

Article history:
Received 17 August 2022
Revised 21 March 2023
Accepted 28 March 2023
Available online 31 March 2023

MSC:
0000
1111

Keywords:
Anomaly detection
One-class classification
GAN
Mode collapse
Cyber security

ABSTRACT

Cyber attacks and anomaly detection are problems where the data is often highly unbalanced towards normal observations. Furthermore, the anomalies observed in real applications may be significantly different from the ones contained in the training data. It is, therefore, desirable to study methods that are able to detect anomalies only based on the distribution of the normal data. To address this problem, we propose a novel objective function for generative adversarial networks (GANs), referred to as STEP-GAN. STEP-GAN simulates the distribution of possible anomalies by learning a modified version of the distribution of the task-specific normal data. It leverages multiple generators in a step-by-step interaction with a discriminator in order to capture different modes in the data distribution. The discriminator is optimized to distinguish not only between normal data and anomalies but also between the different generators, thus encouraging each generator to model a different mode in the distribution. This reduces the well-known *mode collapse* problem in GAN models considerably. We tested our method in the areas of power systems and network traffic control systems (NTCSs) using two publicly available highly imbalanced datasets, ICS (Industrial Control System) security dataset and UNSW-NB15, respectively. In both application domains, STEP-GAN outperforms the state-of-the-art systems as well as the two baseline systems we implemented as a comparison. In order to assess the generality of our model, additional experiments were carried out on seven real-world numerical datasets for anomaly detection in a variety of domains. In all datasets, the number of normal samples is significantly more than that of abnormal samples. Experimental results show that STEP-GAN outperforms several semi-supervised methods while being competitive with supervised methods.

1. Introduction

Cyber attack detection, as well as anomaly detection, have become increasingly important with the digitization of most of our critical infrastructure. Machine learning methods provide a viable alternative to solving these problems. Most studies that attempt to detect cyber attacks use some form of supervised learning, assuming the nature of the attacks is known [1–5]. However, collecting attack data is a challenging process, and attackers, with the advancement of technology, are able to use a variety of sophisticated methods to perform innovative cyber attacks, making it difficult to predict the nature of the attacks. Furthermore, labeled data is not always available in real-world applications, and data labeling is a costly and time-consuming process, requiring the use of human resources that can be associated with human error. Similarly, in the more general area of anomaly detection, it is unrealistic to assume that the kind of anomalies that may occur in the future are known in advance. Supervised methods have a tendency to become highly optimized to the anomalies encountered in the specific datasets. Therefore, in spite of high performance on the specific dataset, these methods may be prone to failure if the characteristics of the anomalies change at a future time.

A solution is to cast the anomaly detection problem into a one-class classification problem where only normal data is used during training. The attack data (anomalies) is assumed to belong to the *complement distribution* [6–8] of normal data. In simple terms, the complement distribution has probability mass in regions where the data distribution has low probability mass, with some extra constraints to ensure that the total probability is bounded. Because no complement data samples are available during

https://doi.org/10.1016/j.neucom.2023.03.056
0925-2312/© 2023 The Author(s). Published by Elsevier B.V.

training, methods that fall into this category are usually generative in nature. Among these methods, generative adversarial networks (GANs) have been a popular choice for this problem [9–11]. However, GANs were originally designed to generate samples according to the same distribution of the training data. In order to adapt the GAN framework to this task, for example, an encoder can be added and the reconstruction error can be used to discriminate between normal and abnormal data [9,10]. Another possibility is to try to explicitly model the complement distribution as in OCAN [11] and BAD-GAN [6].

The aforementioned methods are inefficient in practice and suffer from the fundamental problems of regular GANs, i.e., mode collapse. Mode collapse [12] refers to a problem in which the generator only learns to generate artificial data from a few modes of the data distribution missing other modes despite these being well represented in the training data. This problem limits the ability of GAN-based methods to model and detect possible anomalies.

In [13], we proposed STEP-GAN, a novel GAN-based countermeasure to detect anomalies. This method falls under the category of one-class classifiers previously described, in that it is only trained using normal data. Differently from OCAN, which aims at estimating the complement distribution explicitly, STEP-GAN aims to find the boundary between normal and attack data by estimating a perturbed version of the normal data distribution. Moreover, STEP-GAN provides a new technique for contrasting the mode collapse problem by attempting to model all the modes in which attacks (anomalies) can occur. As a result, our system is potentially more robust to unseen attacks. In this study, we extend the results obtained in [13] in several ways. We discuss the principles behind STEP-GAN more thoroughly. We provide a detailed analysis of the mode collapse problem and how STEP-GAN mitigates it in the context of cyber security. Finally, we provide results on real-world anomaly detection problems, comparing to several other methods proposed in the literature.

Our contributions can be summarized as follow:

- We propose the STEP-GAN architecture for anomaly detection tasks giving a detailed explanation of the motivation for its objective function;
- We provide experimental evidence in the cyber-security domain using two well-known, highly imbalanced, publicly available datasets, namely ICS and UNSW-NB15, showing that:
  - STEP-GAN significantly outperforms the state-of-the-art systems in terms of accuracy and F-measure;
  - STEP-GAN performance is stable across domains, namely power systems (ICS) and network traffic control systems (UNSW-NB15);
- A thorough analysis showing that the proposed objective function considerably reduces the *mode collapse* issue in GANs;
- A demonstration that simply adding generations in a multi-generator version of the OCAN technique, MG-OCAN, would not address the mode collapse problem;
- A proof of the generality of our solution by reporting experiments on seven real-world numerical datasets for anomaly detection from a variety of domains. In all datasets, the number of normal samples is significantly more than that of abnormal samples.

In addition, because STEP-GAN does not rely on labeled attack data for the training process, we speculate that this method may be more robust than supervised methods to unseen attacks in real applications.

The rest of this work is organized as follows. First, we discuss the related studies in Section 2. Then, we present the proposed method in Section 3. Subsequently, we give a brief explanation of the experimental setup in Section 4. Results and conclusions are given in Section 5 and 6, respectively.

2. Related works

Many anomaly detection techniques have been developed in recent years. In this section, we will try to give a comprehensive review, with special emphasis on those methods that are closely related to ours or that have been tested on the data that we use in our experiments.

In the domain of cyberattack detection in power systems, Kravchik and Shabtai [14] applied 1D convolutional neural networks (CNN) and autoencoders on both the time and frequency domains, reporting meaningful results on the Industrial Control System (ICS) data [5] (see Section 4). In [15], a conditional GAN (CGAN) model was employed to deploy cyber physical production systems robust to cross-domain attacks.

Wang et al. [16] employed a neural network based model along with an automatic behavioral abstraction technique (ABATe) to detect anomalies in cyber-physical systems. ABATe models relationships between event vectors from normal data available in abundance with cyber-physical systems (CPS). Then the abstract model is used to detect anomalies. Jahromi et al. [4] employed autoencoders (AE) [17] to extract meaningful features from the power system data. Then, several traditional machine learning based classifiers were employed on the ICS dataset, and a 92.47% accuracy was achieved using the gradient boosting (GB) method and learned features. For the original features, a 95.77% accuracy was obtained using ANNs. Hassan et al. [18] combined a random subspace (RS) method with a random tree (RT) classifier, named RSRT, and delivered an ensemble of trees, which allowed to reduce redundancy of features and prevent overfitting. Results showed that the proposed solution achieved a high attack detection rate (95.95% accuracy) on the ICS dataset. In [19], the authors modeled the dynamic interactions of industry 4.0 [20] components, including a smart management module and a threat intelligence module. The former module handles heterogeneous data sources; whereas the latter module is designed based on beta mixture-hidden Markov models (MHMMs) for the detection of anomalies in both physical and network systems. Accuracy of 98.45% on the ICS dataset.

The authors of [19] also report 96.32% accuracy on the UNSW-NB15 dataset [21], which is in the domain of network traffic control systems. In the same domain, in [22], a deep stacked autoencoder (DSAE) with a softmax classifier was devised to tackle the anomaly detection problem, achieving an 89.13% accuracy on the UNSW-NB15 dataset. Rabbani et al. [23] applied a particle swarm optimization-based probabilistic neural network (PSO-PNN) for the detection and recognition process. In their first module, the user behaviors are converted into an understandable format and then classified by an ANN. A 96.4% detection rate was reported on the UNSW-NB15 dataset. In [24], the averaged one-dependence estimator (AODE) technique was adopted as the basic block of the proposed multiple class classifier, and an accuracy of 83.47% was delivered on the UNSW-NB15 dataset. On the same dataset, a random forest (RF) classifier was employed in [25] attaining an accuracy of 81.62%. In [26], a feature reduction method leveraging two types of pigeon inspired optimizer (PIO) were used: sigmoid PIO and cosine PIO. The sigmoid PIO selected 14 features; whereas the cosine PIO selected 5 features. On the UNSW-NB15 dataset, the Sigmoid PIO achieved an accuracy of 91.3%. In contrast, Cosine PIO obtained 91.7%.

Recently, Zheng et al. [11] proposed one-class adversarial nets (OCAN) for fraud detection on online applications such as social media. OCAN employs an LSTM-Autoencoder [27] to learn the representations of benign users from their sequences of online activities. It then detects malicious users by training a discriminator of a complement GAN model. Whereas the generator in regular GAN learns the distribution of normal data, the generator in complement GAN is trained to generate a distribution of data that is close to the *complement distribution* of the normal data. The concept of complement distribution was introduced in BAD-GAN [6] and was defined as the reciprocal of the normal data distribution if the latter exceeds a certain threshold and as a constant otherwise. To ensure bounded probabilities, the data space is supposed to be bounded by a convex set. Fig. 1 illustrates the difference between regular GAN and BAD-GAN. According to the BAD-GAN paper, the complement distribution p^* in OCAN is defined as follows

$$p^*(x) = \begin{cases} \frac{1}{\tau}\frac{1}{p(x)}, & \text{if } p(x) > \epsilon \text{ and } x \in B_x \\ C, & \text{if } p(x) \leqslant \epsilon \text{ and } x \in B_x, \end{cases} \tag{1}$$

where $p(x)$ is the distribution of the normal data, τ is a normalizer, C is a small constant, ϵ is a hyper-parameter, and the normal data $x \sim p(x)$ is bounded by convex set B (i.e B_x). Accordingly, the KL-divergence between the generator G (with the corresponding distribution p_G) and p^* in OCAN is

$$KL(p_G||p^*) = -H(p_G) + \mathop{\mathbb{E}}_{x\sim P_G} \log \mathrm{II}[p(x) > \epsilon] + \mathop{\mathbb{E}}_{x\sim p_G}(\mathrm{II}[p(x) > \epsilon] \log \tau - \mathrm{II}[p(x) \leqslant \epsilon] \log C), \tag{2}$$

where $H(.)$ is the cross-entropy, $\mathrm{II}[.]$ is an indicator function. Also, the discriminator D in OCAN is a standard feedforward neural network that uses the softmax function as an output layer which maximizes the following objective function

$$\mathop{\mathbb{E}}_{x\sim p(x)}[\log D(x)] + \mathop{\mathbb{E}}_{z\sim p(z)}[\log(1 - D(G(z)))] + \mathop{\mathbb{E}}_{x\sim p(x)}[D(x)\log D(x)], \tag{3}$$

where $p(z)$ is the noise distribution, and $z \sim p(z)$ is the input of the generator. In Eq. 3 the first two terms are the objective function of the discriminator in the standard GAN. Thus, the discriminator of OCAN is trained to discriminate the normal data and complement data. The third term in Eq. 3 is a conditional entropy term, pushing the discriminator to detect normal samples with high confidence. Similar to most GAN based methods, OCAN suffers from mode collapse.

In our experiments, we will compare our results with those reported by the above mentioned works. We also include a detailed comparison with OCAN which inspired our model. Because OCAN was never tested on the ICS, UNSW-NB15, and real-world datasets, we re-implemented the method. Moreover, to make the comparison fairer, we propose and implemented a multi-generator version of OCAN, which we call MG-OCAN. With the help of MG-OCAN, we show how the STEP-GAN optimality criterion is superior to OCAN's in avoiding the mode-collapse problem, even when both methods use multiple generators.

3. Proposed method

In this work, we propose a GAN-based model, which we call STEP-GAN, with a novel objective function, with the goal of detecting anomalies (attacks) when only normal data is available for training. In a regular GAN, a generator $G(z;\theta_g)$ produces fake samples according to a distribution p_g with parameters θ_g given random samples $z \sim p(z)$ as input. A discriminator $D(x;\theta_d)$ tries to distinguish between these fake samples and the real data $(x^n_{i=1}) \sim p_d$. In the GAN objective, however, the adversarial optimization between generator and discriminator is used to make p_g a close approximation of the distribution of the real data p_d [28]. Consequently, in the original formulation, GANs are not suitable for anomaly detection tasks. Moreover, GANs are affected by a number of issues, such as i) mode collapse (Section 1), ii) the need for large amounts of training data, and iii) difficult optimization.

STEP-GAN attempts to generate fake data by estimating a perturbed version of the real data distribution. This is achieved by making the optimization of the generator dependent on the performance of the discriminator during training. Furthermore, this approach allows us to extend the model to incorporate multiple generators. The objective function ensures that different generators will tend to model different nodes in the data distribution, thus reducing the mode collapse problem.

The overall architecture of the proposed system is shown in Fig. 2. The model involves n generators and one discriminator. The discriminator $D(x;\theta_d)$ assigns each observation x to one of $n+1$ classes determining if the observation has been generated by one of the n generators or belongs to the real data (class $n+1$).

To train our system, we first apply noise $z \sim p(z)$ to the input of the multiple generators. Each generator i outputs a fake sample $\tilde{x}_i = G_i(z;\theta^i_g)$ according to the distribution p_{g_i}. The parameters θ^i_g of each generator are optimized by minimizing

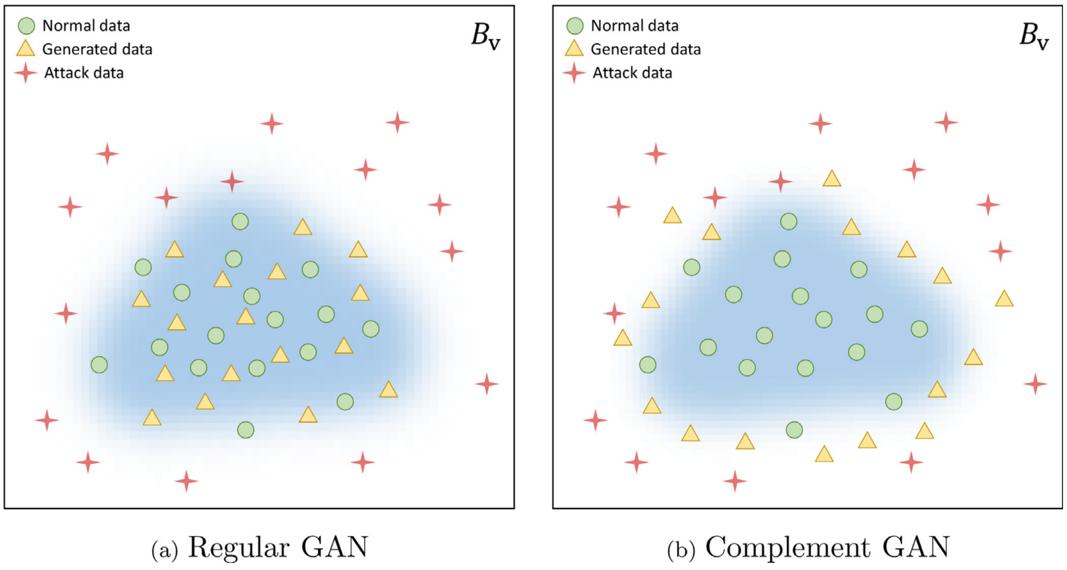

Fig. 1. A comparison of data generation in regular and complement GAN. The blue regions indicate the high density regions of the normal data.

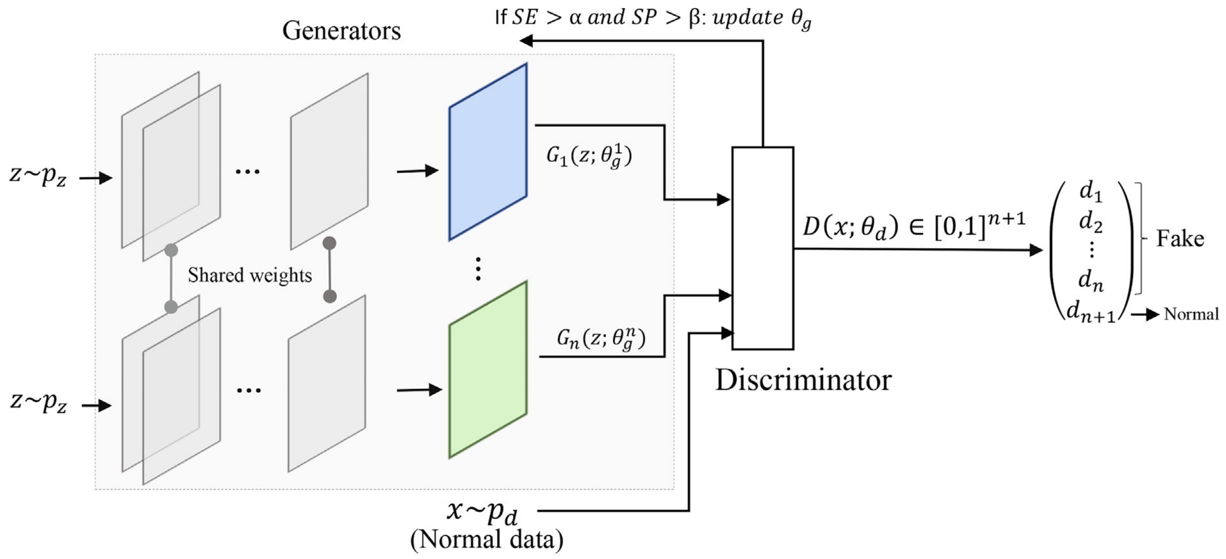

Fig. 2. The architecture of the proposed system with n generators. The discriminator returns $n+1$ softmax values determining the probability of the input data belonging to either one of the the n generators or the normal data.

$$\mathop{\mathbb{E}}_{z\sim p(z)} \log(1 - D_{n+1}(G_i(z;\theta_g^i);\theta_d)), \tag{4}$$

which corresponds to the expected value with respect of the noise distribution $p(z)$ of the log probability that the generated sample $\tilde{x}_i$ is classified as fake by the discriminator.

The parameters of each generator are updated by computing the gradient

$$\nabla_{\theta_g^i} \mathop{\mathbb{E}}_{z\sim p(z)} \log(1 - D_{n+1}(G_i(z;\theta_g^i);\theta_d)). \tag{5}$$

Note that all generators, in this case, can be updated simultaneously. The joint objective of all the generators is to minimize

$$\sum_{i=1}^{n} \mathop{\mathbb{E}}_{x\sim p_{g_i}} \log(1 - D_{n+1}(x)). \tag{6}$$

where we have expressed the expectation in terms of the generated samples x instead of the input samples z.

If we keep the parameters θ_g^i of the generators constant, the theoretically optimal distribution for the discriminator D has the following form:

$$D_i(x) = \begin{cases} \frac{1}{N(x)} p_{g_i}(x), & i \in [1,\dots,n] \\ \frac{1}{N(x)} p_d(x), & i = n+1, \end{cases} \tag{7}$$

with

$$N(x) = \quad p_d(x) + \sum_{j=1}^{n} p_{g_j}(x) \tag{8}$$

where, $D_i(x) \in [0,1], \sum_{i=1}^{n+1} D_i(x) = 1$ represents the ith index of $D(x;\theta_d)$, and p_{g_i} the distribution of the ith generator given θ_g^i. If we consider a sample x (real or fake) to be distributed according to a distribution p with $Supp(p) = \cup_{i=1}^{n} Supp(p_{g_i}) \cup Supp(p_d)$, we can approximate the optimal discriminator by maximizing the expected value of the negative cross entropy function

$$\mathop{\mathbb{E}}_{x\sim p} H(\xi, D(x,\theta_d)), \tag{9}$$

where $\xi \in \{0,1\}^{n+1}$ is the indicator function that specifies if a sample is generated by the ith generator ($i \in \{1,\dots,n\}$) or belongs to the real data ($\xi(n+1) = 1$). This encourages the discriminator to push different generators towards different identifiable modes in order to accurately recognize which generator produced a given fake data point. This phenomenon is responsible for the reduction of the mode collapse problem in the proposed method that will be demonstrated experimentally.

For the discriminator, given $x \sim p$ (real or fake) and the corresponding ξ, the gradient is $\nabla\theta_d \log D_j(x;\theta_d)$, where $D_j(x;\theta_d)$ is the j-th index of $D(x;\theta_d)$ for which $\xi(j) = 1$. Therefore, using this approach requires very minor modifications to the standard GAN optimization algorithm and can be easily used with different variants of GAN.

Finally, in order to lead the generators to generate data in a perturbed version of the distribution of normal data, we apply a condition on the min–max interaction between generators and discriminators. The generators continue learning as long as the sensitivities (SEs) and specificities (SPs) [29] of the discriminator are above the values of two hyper-parameters α and β, respectively. When these values fall below the thresholds α or β, we pause training for the generators until the discriminator learns to perform better than those threshold values. The complete objective function can be expressed as:

$$\begin{aligned}\min_{\theta_g(\mathrm{SE}>\alpha,\mathrm{SP}>\beta)} \max_{\theta_d} V(\theta_d,\theta_g) := & \mathop{\mathbb{E}}_{x\sim p_d} \log D_{n+1}(x) \\ & + \sum_{i=1}^{n} \mathop{\mathbb{E}}_{x\sim p_{g_i}} \log(1 - D_{n+1}(x)) \\ & + \mathop{\mathbb{E}}_{x\sim p} H(\xi, D(x,\theta_d)).\end{aligned} \tag{10}$$

This is fundamentally different from estimating the complement distribution explicitly (OCAN, BAD-GAN) as illustrated in Fig. 3. Our goal is not to estimate a distribution of abnormal data but rather to estimate a perturbed version of the distribution of real data that allows us to find good boundaries between normal data and potential anomalies. The pseudo-code of the training and testing phases of the proposed method is illustrated in Algorithm 1 and Algorithm 2, respectively.

Fig. 4 illustrates the effect of our objective function in the case of two generators and two modes in the training data. The generators are pushed by the discriminator toward the different modes. Furthermore, the generated data spans wider support than the real data, allowing the discriminator to estimate boundaries between normal data and potential anomalies.

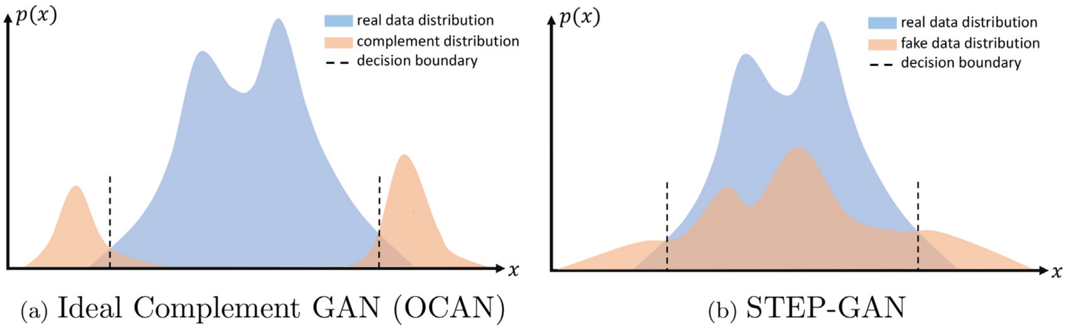

Fig. 3. Illustration of the difference between Complement GAN (OCAN, BAD-GAN) and STEP-GAN. In both cases the real data distribution is shown in blue. a) OCAN estimates the complement distribution to find the optimal discriminator. b) STEP-GAN estimates a perturbed version of the real data distribution.

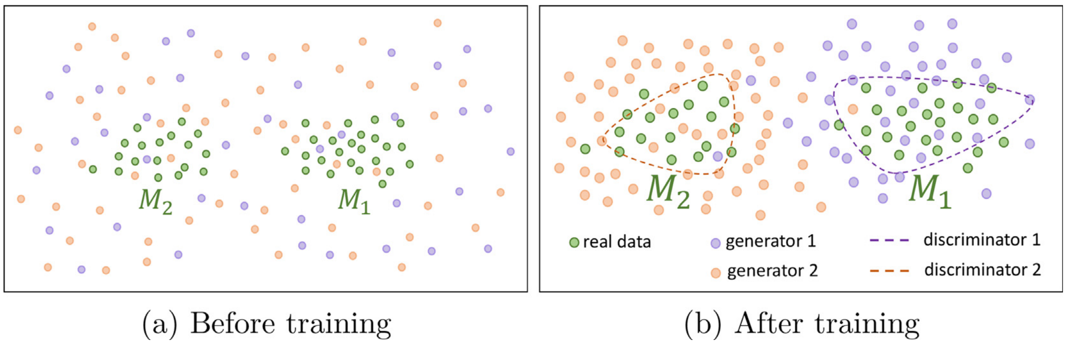

Fig. 4. Visualization of multiple generators that are driven towards different identifiable modes (M_1 and M_2) during the training cycle. After convergence, the discriminators are also shown.

Algorithm 1 STEP-GAN training algorithm.

```
Inputs:
training data $X = \{x_1, \dots, x_N\}$, ▷only normal data
number of generators $n$,
training parameters $\alpha, \beta$,
prior noise distribution $p(z)$,
maximum number of epochs MaxEpochs, and BatchSize
procedure STEP-GAN-Training $X, n, \alpha, \beta, p(z)$
  Initialize model parameters: $\theta_d, \theta_g^i, i \in [1, n]$
  while (Epoch Number < MaxEpochs) do
    for $i \in [1, n]$        ▷for each generator
    Generate BatchSize/$n$ fake samples $G_i(z, \theta_g^i)$ with $z \sim p(z)$
    end for
    Combine fake samples and a batch from $X$ into training batch $B$
    Compute SE and SP over $B$ using current discriminator $\theta_d$
    Update $\theta_d$ using training batch $B$ and $\nabla_{\theta_d} \log D(G_i(z; \theta_g^i), \theta_d)$
    if SE > $\alpha$ and SP > $\beta$
      For each generator $G_i$, update $\theta_g^i$ using $\nabla_{\theta_g^i}(\log(1 - D(G_i(z; \theta_g^i))))$
    end if
    Increment Epoch Number
  end while
  Return trained $\theta_d, \theta_g^i, i \in [1, n]$
end procedure
```

Algorithm 2:STEP-GAN testing algorithm.

```
Inputs:
Number of generators $n$,
Model parameters $\theta_d, \theta_g^i, i \in [1, n]$,
Test data $\mathcal{D} = \{(x_1, t_1), \dots, (x_M, t_M)\}$ ▷both normal and attack data
procedure STEP-GAN-Test$n, \theta_d, \theta_g^i, \mathcal{D}$
  for $x \in \mathcal{D}$ do
    Discriminate normal data from attack data by: $D(x)$ outputting $n+1$ softmax values specifying the probability of the input data belongs to the fake distribution (1 to $n$) or the real distribution ($n+1$).
  end for
end procedure
```

3.1. Assumptions and possible limitations

In the proposed solution, we assume that future normal data will be distributed according to the distribution of the normal data in the training set. If the characteristics of normal data shift in time, some form of adaptation will be required to make our method work, similarly to any other machine learning method. Furthermore, we assume that generators that are not fully trained will model a distribution having roughly the same support as the data distribution but with longer tails (see Fig. 3 (b)). If this is not the case, our method may place the boundaries between normal and attack data in an unreliable manner. However, Fig. 9 (d) shows experimental support for this assumption.

4. Experimental setup

We conducted experiments in the cyber-security domain using two well-known open-source datasets: ICS [5] and UNSW-NB15 [21]. Furthermore, in order to prove the generality of our model, we also conducted experiments on seven real-world numerical datasets for anomaly detection in several domains. These datasets are listed in DevNet paper [30] and are characterized by having a significantly higher number of normal samples compared to the anomalies. We will first describe the data and then give details on the experimental setup.

4.1. The ICS dataset

The ICS dataset was obtained from supervisory control and data acquisition (SCADA) power systems provided by Mississippi State University. The dataset contains three groups, including Binary, Three-Class, and Multiclass datasets. Each group is made from one initial dataset, including 15 subsets that consist of 37 power system event scenarios, comprising 8 *Normal Events*, 1 *No Events*, and 28 *Attack Events*. Different possible types the attack scenarios are listed as follows: *Remote tripping command injection*: in this attack, a command is sent to a relay in order to open a breaker. Such an attack happens when an attacker has infiltrates outside defenses. *Relay setting change*: relays are configured with a distance protection scheme, and the attacker changes the setting to disable the relay function such that the relay will not trip for a valid fault or a valid command. *Data Injection*: attackers try to change values to various parameters such as current, voltage, and sequence components to imitate a valid fault. Such an attack intends to blind the operator, which results in a blackout. The benchmark distribution of each dataset instance is shown in Fig. 5.

Each observation includes 128 fixed-length dimensional features obtained from the phasor measurement units (PMUs). A PMU estimates the magnitude and phase of an electrical current or voltage. These features include 29 types of measurements, containing a total of 116 synchronized phasors (synchrophasor) measurement columns. The synchrophasor, or PMU is built upon the cyber layer and provides real-time data to the energy management system (EMS) in order to control the physical system. Such processes are presented as a sequence of execution events in the cyber-physical environment. The synchrophasor data comprises not only the measurements (e.g., voltage and current phasors) but also the status of system devices consisting of relays, breakers, switches, and transformers. Furthermore, there are 12 types of measurements of control panel logs, snort alerts, and relay logs of the 4 synchrophasor measurement unit and relay.

In this study, we used binary classification events for detecting false data injection attacks on the SCADA system. In order to reduce the effect of small sample sizes, the datasets were randomly sampled at 1% to reduce the size and evaluate the effectiveness of small sample sizes. The dataset statistics of binary class events classification are summarized in Table 1.

4.2. The UNSW-NB15 dataset

There are many publicly available datasets on network intrusion detection systems (NIDSs). The most widely used among them are DARPA 98 [31], KDD Cup 99 [32], NSL-KDD [33]. DARPA 98 contains several weeks of network data and audit logs, but this dataset does not depict real-world traffic. KDD Cup 99 was published to improve the problems of DARPA 98. However, this dataset contains

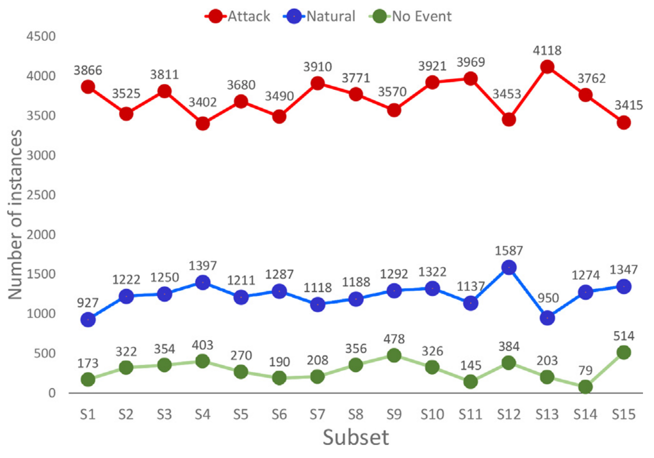

Fig. 5. Distribution of benchmark ICS dataset.

Table 1
The ICS dataset statistics of binary classification events.

Subset	#Event Scenario	#Instances
No Events	1	294
Natural Event	8	1221
Attack	28	3711
Total	37	5226

issues such as duplicate and redundant records. NSL-KDD dataset was released to refine KDD 99 problems. However, many studies have shown that NSL-KDD does not inclusively reflect network traffic and modern low footprint attacks of real environment [21]. UNSW-NB15 was introduced in 2015, claiming to contain the most comprehensive attack scenarios. The UNSW-NB15 dataset was obtained from 9 types of modern attack methods, and new patterns of normal network traffic flows. The traffic generator tool, called IXIA traffic generator, is utilized as an attack traffic generator along with as normal traffic. The attack behavior is nourished from the CVE site[1] for the purpose of a real representation of a modern threat environment. Due to the speed of network traffic and the way of exploiting by modern attacks, the IXIA tool is configured to generate one attack per second during the first simulation to capture the first 50 GBs. On the other hand, the second simulation is configured to make ten attacks per second to extract another 50 GBs. The dataset includes 49 features, containing the flow based between hosts (i.e., client-to-server or server-to-client) and the network packets inspection to distinguish between the normal or abnormal samples [21]. The list of features is reported in Table 2. Flow-based features are generated using the sequencing of packets from a source to a destination, traveling in the network. In contrast, Packet-based features are extracted from the packet header and its payload (also called packet data). The features are classified into 3 sets, called basic features (6 to 18), content features (19 to 26), and time features (27 to 35). In addition, features 36 to 40 and 41 to 47 are labeled as general-purpose features and connection features, respectively. Features in the dataset are represented in both quantitative (i.e., numeric) and qualitative (i.e., symbolic) types. Since the model can process only quantitative data, we used a unified format to convert all non-quantitative features into numeric ones like [23]. Details of UNSW-NB15 are summarized in Table 3.

In Fig. 6, we visualize the training sets in UNSW-NB15 and ICS-S1 by means of t-SNE dimensionality reduction. As can be seen, attack data are more prevalent in regions where normal data has

[1] https://cve.mitre.org/

Table 2
Features in the UNSW-NB15 dataset.

Category	No.	Name	Data Type	Category	No.	Name	Data Type
Flow	1	scrip	nominal	**Content**	25	trans_depth	Integer
	2	sport	Integer		26	res_bdy_len	Integer
	3	dstip	Nominal	**Time**	27	Sjit	Float
	4	dsport	Integer		28	Djit	Float
	5	proto	Nominal		29	Stime	Timestamp
Basic	6	state	Nominal		30	Ltime	Timestamp
	7	dur	Float		31	Sintpkt	Float
	8	sbytes	Integer		32	Dintpkt	Float
	9	dbytes	Integer		33	tcprtt	Float
	10	sttl	Integer		34	synack	Float
	11	dttl	Integer		35	ackdat	Float
	12	sloss	Integer	**General**	36	is_sm_ips_ports	Binary
	13	dloss	Integer	**Purpose**	37	ct_state_ttl	Integer
	14	service	nominal		38	ct_flw_http_mthd	Integer
	15	Sload	Float		39	is_ftp_cmd	Binary
	16	Dload	Float		40	ct_fto_cmd	Integer
	17	Spkts	Integer	**Connection**	41	ct_srv_src	Integer
	18	Dpkts	Integer		42	ct_srv_dst	Integer
Content	19	swin	Integer		43	cv_dst_ltm	Integer
	20	dwin	Integer		44	ct_src_ltm	Integer
	21	stcpb	Integer		45	ct_dst_dport_ltm	Integer
	22	dtcpb	Integer		46	ct_dst_sport_ltm	Integer
	23	smeansz	Integer		47	ct_dst_src_ltm	Integer
	24	dmeansz	Integer		48	attack_cat	Nominal
					49	class	Binary

Table 3
The UNSW-NB15 dataset statistics and descriptions of binary classification events.

Subset	#Training set	#Testing set	Description
Normal	56000	37000	Natural transaction data.
Analysis	2000	677	Contains different attacks of port scan, spam and html files penetrations.
Backdoor	1746	583	A technique in which a system security mechanism is bypassed stealthily to access a computer or its data.
DoS	12264	4089	Attempts to make a server or a network resource unavailable to users, usually by temporarily interrupting or suspending the services of a host connected to the Internet.
Exploits	33393	11132	The attacker knows of a security problem within an operating system or a piece of software and leverages that knowledge by exploiting the vulnerability.
Fuzzers	18184	6062	Attempting to cause a program or network suspended by feeding it the randomly generated data.
Generic	40000	18871	A technique works against all blockciphers (with a given block and key size), without consideration about the structure of the block-cipher.
Reconnaissance	10491	3496	Contains all Strikes that can simulate attacks that gather information.
Shellcode	1133	378	A small piece of code used as the payload in the exploitation of software vulnerability.
Worms	130	44	Attackers try to iterate themselves to spread to other systems. These types of attack mostly use a computer network to spread itself, based on security failures on the target computer to access it.
Total	175341	82332	–

a low probability of existing, while its distribution is very similar to normal data distribution, i.e., complement distribution. Note that only the normal data is used for training our model.

4.3. Real-world datasets

The seven real-world datasets used in this work are: Annthyroid, Campaign, Celeba, Fraud, Donors, Backdoor, and Census. Those datasets cover a wide variety in sample size, feature dimensionality, and anomaly ratio in the real world, which provide a suitable benchmark to assess the robustness of the tested solution against different types of anomalies. Details for each dataset that are relevant to this study are given in Table 5. More information can be found in the DevNet paper [30].

4.4. Models

Both the generators and the discriminator in the model are composed of fully connected layers. The first layers in the generators share weights in order to reduce computational requirements and speed up convergence. The hyperparameters of the models are optimized for each dataset using an independent evaluation set. Details about the configuration of the architecture for both ICS and UNSW-NB15 datasets are given in Table 4. The configuration used for UNSW-NB15 was employed for real-world datasets. The only difference is the input size, which depends on the dimension of the input data.

For all configurations, we use the Adam optimizer [35] for training generators and the discriminator. The cross-entropy is also used as a loss function. The main experimental parameters that are varied in our experiments are the number of generators in the model that can assume any value in $\{1, 2, 3, 5, 10, 15, 20\}$ and the hyper-parameters α and β that are varied between 0.55 and 1.0 in intervals of 0.05, for both ICS and UNSW-NB15 datasets. With a real-world dataset, we trained our model using 10 generators, and α and β were set equal to 0.9.

4.5. Evaluation metrics

Anomaly detection is a binary classification task. Therefore, to verify the performance of the proposed method we used three metrics: Accuracy, F-measure, receiver operating characteristic

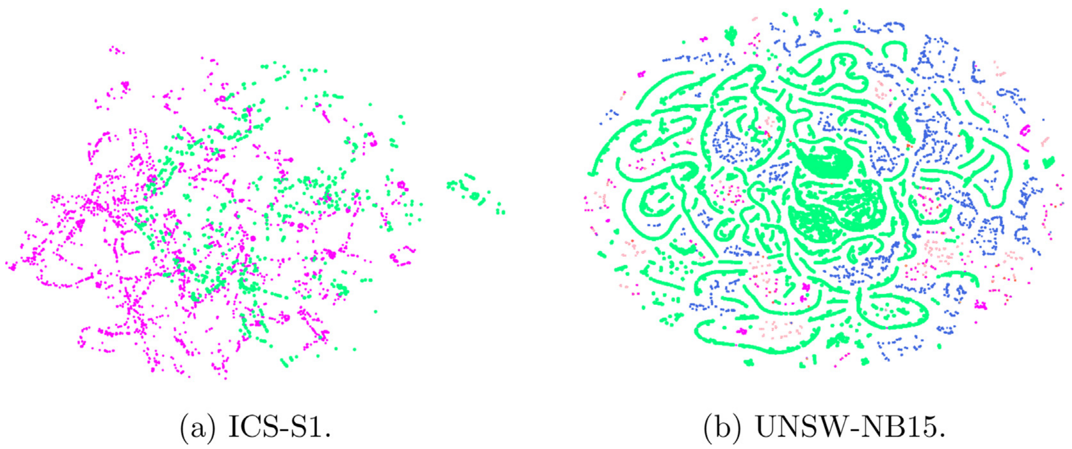

Fig. 6. 2D t-SNE visualization of training sets of the datasets. Green points indicate normal data, and other points indicate different types of attack data. Note that only normal data is used to train our model.

Table 4
Architecture details. n is the number of generators.

		Input	Hidden			Output	
Data	Model	nodes	layers	nodes	activation	nodes	activation
ICS	generator	50	3	300	Parametric ReLU [34]	128	tanh
	discriminator	128	4	300	Leaky ReLU	$n+1$	softmax
UNSW-NB15	generator	20	4	250	Parametric ReLU	49	tanh
	discriminator	49	3	300	Leaky ReLU	$n+1$	softmax

Table 5
The statistics of the real world dataset.

Name	Data size	Dimensionality	# Anomaly Ratio	Category
Annthyroid	7,200	21	7.42%	Healthcare
Campaign	41,188	62	11.27%	Finance
Celeba	202,599	39	2.23%	Image
Fraud	284,807	29	0.17%	Finance
Donors	619,326	10	5.93%	Sociology
Backdoor	95,329	196	2.44%	Network
Census	299,285	500	6.20%	Sociology

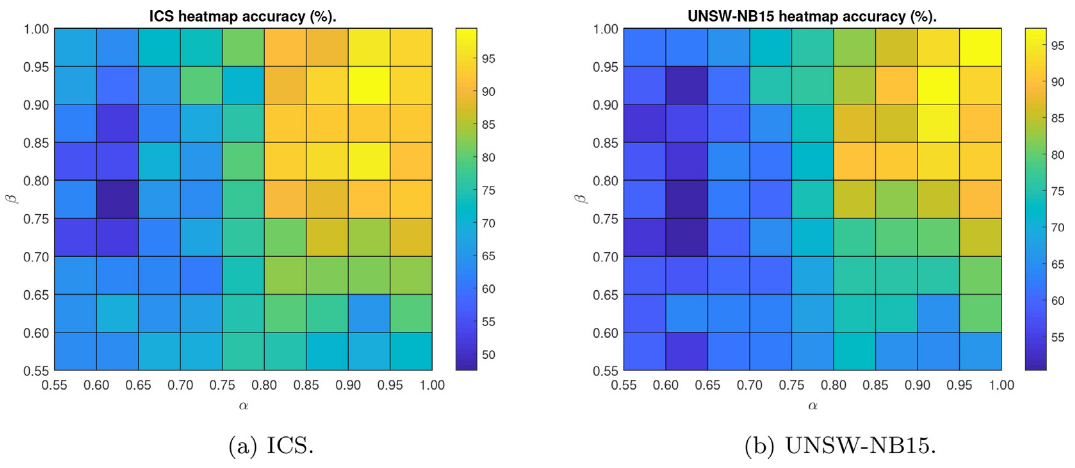

Fig. 7. The heat-map diagram of the accuracies obtained from the proposed model for different values of α and β and a fixed ten generators on (a) ICS and (b) UNSW-NB15 dataset.

Table 6
STEP-GAN accuracy (%) as a function of hyper-parameters on ICS and UNSW-NB15 development sets.

#Generators	Hyper-parameters (α, β)				
	(0.95,0.95)	(0.9,0.9)	(0.8,0.8)	(0.7,0.7)	(0.6,0.6)
			ICS		
1	59.46	64.18	53.84	55.17	49.21
2	81.07	86.34	74.67	53.78	54.23
3	93.34	93.55	73.56	61.45	53.09
5	94.67	97.59	92.31	67.93	65.40
10	95.21	**98.60**	94.88	75.43	74.98
15	87.57	98.44	87.07	64.23	64.94
20	90.49	96.23	83.71	62.53	57.69
			UNSW-NB15		
1	68.45	75.45	71.45	64.15	59.01
2	85.58	92.63	81.46	65.18	57.93
3	96.77	95.05	90.66	73.56	64.17
5	96.23	97.37	93.90	75.89	69.55
10	97.24	**98.45**	95.84	76.57	72.11
15	94.34	97.06	91.20	74.84	62.56
20	90.69	96.42	90.03	78.84	64.67

Table 7
Comparison between MG-OCAN (OCAN) and STEP-GAN depending on the number of generators evaluated on the test sets for the ICS and UNSW-NB15 datasets.

System	#Generators	ICS Dataset		UNSW-NB15 Dataset	
		Accuracy%	F-measure	Accuracy%	F-measure
OCAN	1	63.97	0.2791	75.03	0.6184
MG-OCAN	2	64.70	0.3054	75.12	0.6138
	3	71.36	0.5116	78.95	0.6396
	5	74.08	0.5491	79.38	0.6549
	10	77.53	0.6054	85.03	0.7723
	15	80.14	0.6290	89.31	0.8674
	20	82.56	0.6739	90.62	0.8932
STEP-GAN (ours)	1	65.74	0.6012	76.28	0.6274
	2	88.21	0.8446	90.14	0.8817
	3	92.54	0.9012	95.48	0.9128
	5	98.73	0.9537	96.52	0.9330
	10	**99.51**	**0.9762**	**97.24**	**0.9644**
	15	98.12	0.9495	96.78	0.9348
	20	97.38	0.9375	96.55	0.9317

(ROC) curve, and the area under the curve (AUC) (i.e. AUC-ROC) [36], which is insensitive to the number of outliers:

$$\text{Accuracy} = \frac{T_P + T_N}{T_P + T_N + F_P + F_N}, \tag{11}$$

$$\text{F} - \text{measure} = \frac{2 \times T_P}{2 \times T_P + F_N + F_P}, \tag{12}$$

where T_P, T_N, F_P and F_N indicate *true positive*, *true negative*, *false positive* and *false negative*, respectively.

4.6. Training and test phases

In order to train our model, for the ICS dataset, we randomly selected 80% of the dataset as the training set, 10% for the validation set, and the rest as the test set. For UNSW-NB15, we randomly selected 10% of the training data and used as a validation set. The validation sets were used to fine-tune all of the parameters in our models. Importantly, only the discriminator was used during the testing phase to detect anomalies, that is, malicious attacks. For real-world datasets, we used a 10-fold technique, where each dataset is randomly divided into 90% for training and 10% for testing. In addition, we use 50% abnormal data in training and the remaining 50% in test data.

4.7. Baseline systems

In order to evaluate the performance of our method, we compare STEP-GAN with a complement GAN-based model, namely one-class adversarial networks (OCAN) [11] as a baseline system, which was described in Section 2. In order to have a fair

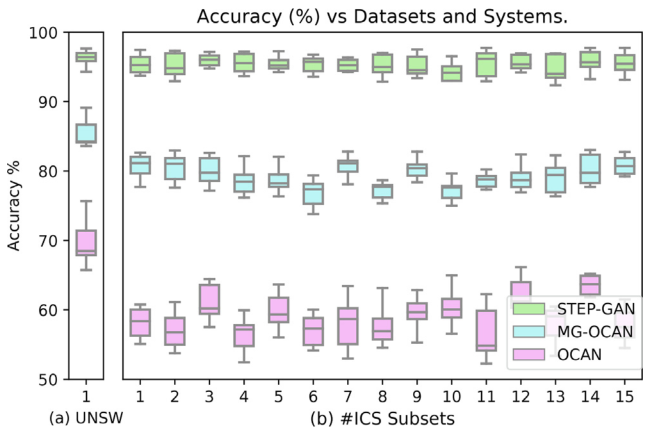

Fig. 8. A comparison of STEP-GAN with baseline systems for UNSW-NB15 test set (a) and each subset of ICS test sets (b) obtained from the best configurations of each system for 5 repetitions.

comparison between STEP-GAN and OCAN, as mentioned in Section 2, we also implemented OCAN with multiple generators (MG-OCAN). We optimized the number of generators in MG-OCAN in a similar way as for STEP-GAN.

5. Results

5.1. Hyper-parameter optimization

The number of generators n, and the thresholds α and β were optimized using an independent development set resulting in an optimum at $n = 10, \alpha = \beta = 0.9$ both for the ICS and for the UNSW-NB15 datasets. Part of this optimization is shown in Fig. 7 where we keep the number of generators constant ($n = 10$) and we make α and β vary with intervals of 0.05. These results show that variations in α values affect the performance of the model more than β variations. This means that the model is more dependent on the performance of the discriminator for detecting normal data (higher value of sensitivity) than the performance of generators for simulating fake data (higher values of specificity).

Table 6, shows the optimization of the number of generators (for some of the values of α and β) on ICS and UNSW-NB15 datasets. These results show that the optimal values of the hyperparameters are consistent across tasks. They also show that performance is strongly dependent on both the number of generators and the values of the hyperparameters. After optimizing hyperparameters for ICS and UNSW-NB15 datasets, those same hyperparameters were used to build our model using real-world data (i.e. $\alpha, \beta = (0.9, 0.9)$, and $n = 10$).

5.2. Comparison with baselines and state-of-the-art

Table 7 shows a comparison between STEP-GAN and the baseline models (OCAN and MG-OCAN) as a function of the number of generators. The evaluation is performed on the ICS and UNSW-NB15 test sets. From the table, it is clear that STEP-GAN

Table 8
Comparison with state-of-the-art on the ICS and UNSW-NB15 test sets.

System	ICS Dataset		UNSW-NB15 Dataset	
	Accuracy%	F-measure	Accuracy%	F-measure
GB (learned features) [4]	92.47	0.9057	–	–
ANN (original features) [4]	95.77	0.9150	–	–
MHMM [19]	98.45	–	96.32	–
RSRT [18]	95.95	–	–	–
PSO-PNN [23]	–	–	–	0.9750
DSAE [22]	–	–	89.13	0.9085
Sigmoid PIO [26]	–	–	91.30	0.9040
Cosine PIO [26]	–	–	91.70	0.9090
STEP-GAN (10 generators)	**99.51**	**0.9762**	**97.24**	0.9644

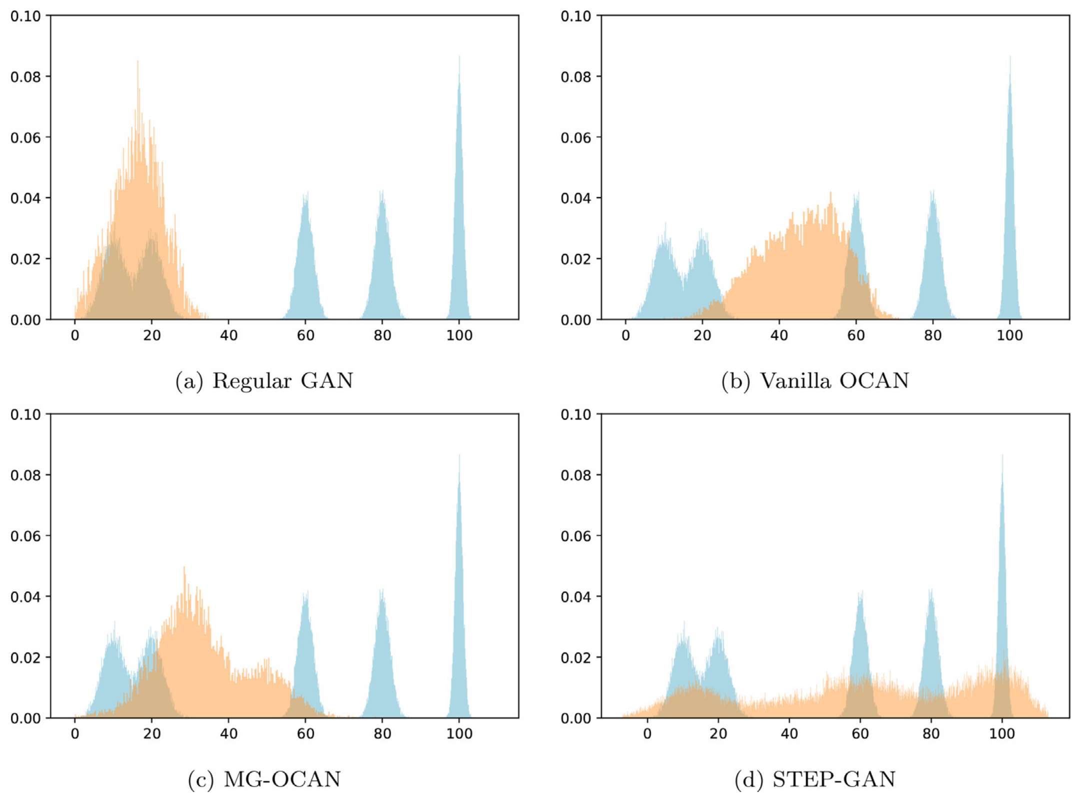

Fig. 9. Simulated example to demonstrate the behavior of the models with respect to model collapse. Each model was trained for 210,000 epochs. Blue histograms correspond to the training (real) data. Orange histograms show the distribution of the generated samples.

Table 9
Average AUC-ROCs ↑ (ranking ↓) obtained by different model on the real-world datasets.

Data	Semi-Supervised							Supervised			
	STEP-GAN	VAE [40]	OCAN [11]	MG-OCAN	MO-GAAL [38]	GANomaly [39]	REPEN [41]	Naive Bayes	SVM	MLP	Random Forest
Annthyroid	0.846(3)	0.840(4)	0.475(11)	0.566(10)	0.690(9)	0.768(8)	0.833(5)	0.801(6)	0.820(7)	0.966(2)	0.993(1)
Campaign	0.879(2)	0.730(7)	0.677(8)	0.693(8)	0.780(6)	0.583(10)	0.579(11)	0.811(4)	0.784(5)	0.829(3)	0.912(1)
Celeba	0.777(6)	0.794(5)	0.641(10)	0.684(9)	0.758(7)	0.698(8)	0.576(11)	0.894(3)	0.880(4)	0.954(1)	0.894(2)
Fraud	0.981(1)	0.968(2)	0.942(5)	0.950(4)	0.955(3)	0.908(7)	0.922(6)	0.885(8)	0.771(11)	0.863(10)	0.881(9)
Donors	0.942(5)	0.820(9)	0.768(10)	0.845(6)	0.830(7)	0.576(11)	0.828(8)	0.996(4)	0.999(2)	1.000(1)	0.998(3)
Backdoor	0.874(6)	0.886(5)	0.517(11)	0.641(10)	0.625(11)	0.864(8)	0.896(4)	0.851(8)	0.962(3)	0.976(2)	0.996(1)
Census	0.794(4)	0.657(9)	0.623(8)	0.665(7)	0.586(10)	0.584(11)	0.694(6)	0.733(5)	0.853(3)	0.856(2)	0.908(1)
avg. AUC	0.871(3)	0.813(6)	0.663(11)	0.720(9)	0.746(8)	0.711(10)	0.761(7)	0.853(5)	0.867(4)	0.922(2)	0.931(1)
avg. rank	3.85	5.85	9.00	7.17	7.57	9.00	7.28	5.42	5.00	3.00	2.57

outperforms MG-OCAN and OCAN on the proposed tasks for all configurations. It is also interesting to point out that, while the accuracy does not degrade when we go past the optimal number of generators in STEP-GAN, the F-measure degrades considerably. This suggests that the method can still correctly predict the dominant class (normal data) but makes more mistakes on the anomalies. The best configuration for STEP-GAN and baseline models are compared in Fig. 8. The boxplots show the results over 5 independent repetitions. This figure reveals how the performance improvement for our system is consistent and stable over repetitions and over UNSW-NB15 and subsets of the ICS databases.

Table 8 summarizes the results with respect to the state-of-the-art on the two tasks we have considered. STEP-GAN achieves the overall best results. It is also worth noting that a suboptimal configuration of STEP-GAN with 5 generators still outperforms most competing systems.

5.3. Mode collapse

To investigate the behavior with respect to mode collapse, we designed a simple simulated experiment. We consider the distribution of one dimensional Gaussian mixture model (GMM) [37] including five mixture components with modes at $\{10, 20, 60, 80, 110\}$, and standard deviations of $\{3, 3, 2, 2, 1\}$, respectively. In this experiment, the first two modes clearly overlap. However, the fifth mode lies separately as depicted in Fig. 9. For MG-OCAN, we used 4 generators, and for STEP-GAN, We trained the model with 4 generators using $\alpha = 0.95, \beta = 0.95$. As shown in Fig. 9, STEP-GAN estimates a distribution that is close to the real data distribution but with broader support. On the other hand, both OCAN and MG-OCAN suffer from mode collapse and cover only a limited domain of complement distributions of existing modes. Although MG-OCAN is trained with 4 generators, it still fails to capture different modes, while can successfully capture all 5 modes. This is mainly due to the STEP-GAN objective function, which pushes multiple generators to generate samples into different identifiable modes.

5.4. Results on real-world datasets

Here we compare the results of STEP-GAN with 6 semi-supervised techniques and 4 supervised methods on seven real-world datasets for anomaly detection [30]. The AUC-ROC results, along with corresponding rankings, are shown in Table 9. The methods listed in Table 9 except OCAN and MG-OCAN are trained using PyOD[2] implementation with the default parameters. STEP-GAN outperforms all GAN-based models (OCAN [11], MG-OCAN, MO-GAAL [38], GANomaly [39]) on all tested datasets. Moreover, STEP-GAN attains better results than variational autoencoder (VAE) [40] on 5 out of 7 datasets and REPEN [41] on 6 out of 7 datasets.

In our opinion, comparing semi-supervised and supervised models in this domain is not entirely fair because the second has an advantage when the test material is not too dissimilar from the training material. However, even though STEP-GAN is a semi-supervised method and only uses normal data in the training process, it still shows better average performance than 2 of the 4 listed supervised methods.

These results prove how STEP-GAN generalizes well across domains and is highly competitive among semi-supervised techniques.

6. Conclusions

In this study, we propose a novel countermeasure to detect cyber attacks, also known as anomalies in the literature, that we call STEP-GAN. STEP-GAN is a multi-generators GAN-based model that utilizes a novel optimality criterion in conjunction with a step-by-step training procedure in order to simulate possible attacks on the system using only normal training data. We report results on two highly imbalanced publicly available ICS and UNSW-NB15 datasets showing that our model significantly outperforms the state-of-the-art on these tasks, as well as OCAN and MG-OCAN that were not previously tested in these domains. We also show that the reason for this performance improvement is that the proposed training procedure mitigates the mode collapse issue in GAN based systems and therefore generates more diverse samples with fewer generators.

Finally, we prove the generality of our method by testing on seven real-life anomaly detection datasets. Here STEP-GAN is very competitive compared to other semi-supervised techniques and even outperforms, on average some supervised techniques.

CRediT authorship contribution statement

Mohammad Adiban: Conceptualization, Methodology, Software, Writing - original draft, Visualization, Investigation. **Sabato Marco Siniscalchi:** Supervision, Writing - review & editing. **Giampiero Salvi:** Supervision, Writing - review & editing.

Data availability

Data is publicly available and the link is mentioned in my paper

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Mohammad Adiban reports financial support, article pub-

[2] https://pyod.readthedocs.io/

lishing charges, and writing assistance were provided by Norwegian University of Science and Technology. Mohammad Adiban reports a relationship with Norwegian University of Science and Technology that includes: employment.

References

[1] J. Sakhnini, H. Karimipour, A. Dehghantanha, Smart grid cyber attacks detection using supervised learning and heuristic feature selection, in: 2019 IEEE 7th International Conference on Smart Energy Grid Engineering (SEGE) IEEE, 2019, pp. 108–112.
[2] F. Zhang, Q. Li, Deep learning-based data forgery detection in automatic generation control, in: 2017 IEEE Conference on Communications and Network Security (CNS) IEEE, 2017, pp. 400–404.
[3] J. Yan, B. Tang, H. He, Detection of false data attacks in smart grid with supervised learning, in: 2016 International Joint Conference on Neural Networks (IJCNN) IEEE, 2016, pp. 1395–1402.
[4] A.N. Jahromi, J. Sakhnini, H. Karimpour, A. Dehghantanha, A deep unsupervised representation learning approach for effective cyber-physical attack detection and identification on highly imbalanced data, in: Proceedings of the 29th Annual International Conference on Computer Science and Software Engineering, 2019, pp. 14–23.
[5] R.C.B. Hink, J.M. Beaver, M.A. Buckner, T. Morris, U. Adhikari, S. Pan, Machine learning for power system disturbance and cyber-attack discrimination, 7th International symposium on resilient control systems (ISRCS), IEEE 2014 (2014) 1–8.
[6] Z. Dai, Z. Yang, F. Yang, W.W. Cohen, R.R. Salakhutdinov, Good semi-supervised learning that requires a bad GAN, in: Advances in neural information processing systems, 2017, pp. 6510–6520.
[7] Y. Xu, M. Gong, J. Chen, T. Liu, K. Zhang, K. Batmanghelich, Generative-discriminative complementary learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 6526–6533.
[8] T. Ishida, G. Niu, A. Menon, M. Sugiyama, Complementary-label learning for arbitrary losses and models, in: International Conference on Machine Learning PMLR, 2019, pp. 2971–2980.
[9] H. Zenati, C.S. Foo, B. Lecouat, G. Manek, V.R. Chandrasekhar, Efficient GAN-based anomaly detection, 2018, arXiv preprint arXiv:1802.06222.
[10] H. Zenati, M. Romain, C.-S. Foo, B. Lecouat, V. Chandrasekhar, Adversarially learned anomaly detection, in: 2018 IEEE International Conference on Data Mining (ICDM) IEEE, 2018, pp. 727–736.
[11] P. Zheng, S. Yuan, X. Wu, J. Li, A. Lu, One-class adversarial nets for fraud detection, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 1286–1293.
[12] A. Srivastava, L. Valkov, C. Russell, M.U. Gutmann, C. Sutton, Veegan: Reducing mode collapse in GANs using implicit variational learning, Adv. Neural Inform. Process. Syst. (2017) 3308–3318.
[13] M. Adiban, A. Safari, G. Salvi, STEP-GAN: A one-class anomaly detection model with applications to power system security, in: ICASSP, IEEE, 2021, pp. 2605–2609.
[14] M. Kravchik, A. Shabtai, Efficient cyber attack detection in industrial control systems using lightweight neural networks and pca, IEEE Trans. Dependable Secure Comput. (2021).
[15] S.R. Chhetri, A.B. Lopez, J. Wan, M.A. Al Faruque, Gan-sec: Generative adversarial network modeling for the security analysis of cyber-physical production systems, Design, Automation & Test in Europe Conference & Exhibition (DATE), IEEE 2019 (2019) 770–775.
[16] S.N. Narayanan, A. Joshi, R. Bose, Abate: Automatic behavioral abstractiontechnique to detect anomalies in smartcyber-physical systems, IEEE Trans. Dependable Secure Comput. (2020).
[17] A. Ng et al., Sparse autoencoder, CS294A Lecture Notes 72 (2011) (2011) 1–19.
[18] M.M. Hassan, A. Gumaei, S. Huda, A. Almogren, Increasing the trustworthiness in the industrial iot networks through a reliable cyberattack detection model, IEEE Trans. Industr. Inf. 16 (9) (2020) 6154–6162.
[19] N. Moustafa, E. Adi, B. Turnbull, J. Hu, A new threat intelligence scheme for safeguarding industry 4.0 systems, IEEE Access 6 (2018) 32910–32924.
[20] F. Zezulka, P. Marcon, I. Vesely, O. Sajdl, Industry 4.0–an introduction in the phenomenon, IFAC-PapersOnLine 49 (25) (2016) 8–12.
[21] N. Moustafa, J. Slay, Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set), Military communications and information systems conference (MilCIS), IEEE 2015 (2015) 1–6.
[22] F.A. Khan, A. Gumaei, A. Derhab, A. Hussain, A novel two-stage deep learning model for efficient network intrusion detection, IEEE Access 7 (2019) 30373–30385.
[23] M. Rabbani, Y.L. Wang, R. Khoshkangini, H. Jelodar, R. Zhao, P. Hu, A hybrid machine learning approach for malicious behaviour detection and recognition in cloud computing, J. Network Comput. Appl. 151 (2020).
[24] M. Nawir, A. Amir, N. Yaakob, O.B. Lynn, Multi-classification of unsw-nb15 dataset for network anomaly detection system, J. Theor. Appl. Inform. Technol. 96 (15) (2018).
[25] T. Janarthanan, S. Zargari, Feature selection in unsw-nb15 and kddcup'99 datasets, IEEE 26th international symposium on industrial electronics (ISIE), IEEE 2017 (2017) 1881–1886.
[26] H. Alazzam, A. Sharieh, K.E. Sabri, A feature selection algorithm for intrusion detection system based on pigeon inspired optimizer, Expert Syst. Appl. 148 (2020).
[27] N. Srivastava, E. Mansimov, R. Salakhudinov, Unsupervised learning of video representations using lstms, in: International conference on machine learning, 2015, pp. 843–852.
[28] A. Ghosh, V. Kulharia, V.P. Namboodiri, P.H. Torr, P.K. Dokania, Multi-agent diverse generative adversarial networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8513–8521.
[29] A.G. Lalkhen, A. McCluskey, Clinical tests: sensitivity and specificity, Continuing Education in Anaesthesia Critical Care & Pain 8 (6) (2008) 221–223.
[30] G. Pang, C. Shen, A. van den Hengel, Deep anomaly detection with deviation networks, in: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, 2019, pp. 353–362.
[31] R.P. Lippmann, D.J. Fried, I. Graf, J.W. Haines, K.R. Kendall, D. McClung, D. Weber, S.E. Webster, D. Wyschogrod, R.K. Cunningham, et al., Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation, in: Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00, vol. 2, IEEE, 2000, pp. 12–26.
[32] M. Tavallaee, E. Bagheri, W. Lu, A.A. Ghorbani, A detailed analysis of the kdd cup 99 data set, in: 2009 IEEE symposium on computational intelligence for security and defense applications, IEEE, 2009, pp. 1–6.
[33] L. Dhanabal, S. Shantharajah, A study on nsl-kdd dataset for intrusion detection system based on classification algorithms, Int. J. Adv. Res. Comput. Commun. Eng. 4 (6) (2015) 446–452.
[34] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 1026–1034.
[35] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.
[36] S. Wu, P. Flach, A scored auc metric for classifier evaluation and selection, in: Second workshop on ROC analysis in ML, bonn, Germany, 2005.
[37] C.M. Bishop, N.M. Nasrabadi, Pattern recognition and machine learning, Vol. 4, Springer, 2006.
[38] Y. Liu, Z. Li, C. Zhou, Y. Jiang, J. Sun, M. Wang, X. He, Generative adversarial active learning for unsupervised outlier detection, IEEE Trans. Knowl. Data Eng. 32 (8) (2019) 1517–1528.
[39] S. Akcay, A. Atapour-Abarghouei, T.P. Breckon, Ganomaly: Semi-supervised anomaly detection via adversarial training, in: Asian conference on computer vision, Springer, 2018, pp. 622–637.
[40] D.P. Kingma, M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114 (2013).
[41] G. Pang, L. Cao, L. Chen, H. Liu, Learning representations of ultrahigh-dimensional data for random distance-based outlier detection, in: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, 2018, pp. 2041–2050.

Mohammad Adiban is a PhD candidate in Machine Learning at the Norwegian University of Science and Technology (NTNU). He received his BS degree in Computer Engineering and received his MS in Artificial Intelligence, from Sharif University of Technology, in 2017. He also visited Monash University in Australia as a researcher in 2022. He is co-founder of the company Connect Me. His research interests span the areas of statistical machine learning, signal processing, computer vision, speech processing, biomedical and cyber security.

Sabato Marco Siniscalchi (Senior Member, IEEE) is a Professor with the University of Enna, Enna, Italy, an Adjunct Professor with the Norwegian University of Science and Technology's (NTNU), and an Affiliate Faculty with the Georgia Institute of Technology. He received his doctorate degree in computer engineering from the University of Palermo, Palermo, Italy, in 2006. In 2006, he was a Postdoctoral Fellow with Ga Tech. From 2007 to 2010, he joined NTNU, Norway, as a Research Scientist. From 2010 to 2015, he was an Assistant Professor, first, and an Associate Professor, after, at the Kore University. From 2017 to 2018, he was a Senior Speech Researcher with Siri Speech Group, Apple Inc., Cupertino CA, USA. He acted as an Associate Editor of the IEEE/ACM Transactions on Audio, Speech and Language Processing, from 2015 to 2019. Prof. Siniscalchi was an Elected Member of the IEEE SLT Committee from 2019 to 2022.

Giampiero Salvi is Professor at the Department of Electronic Systems at the Norwegian University of Science and Technology (NTNU), Trondheim, Norway, and Associate Professor at KTH Royal Institute of Technology, Department of Electrical Engineering and Computer Science, Stockholm, Sweden. Prof. Salvi received the MSc degree in Electronic Engineering from Universitá la Sapienza, Rome, Italy and the PhD degree in Computer Science from KTH. He was a post-doctoral fellow at the Institute of Systems and Robotics, Lisbon, Portugal. He was a co-founder of the company SynFace AB, active between 2006 and 2016. His main interests are machine learning, speech technology, and cognitive systems.