New Perspective on Progressive GANs Distillation for One-class Novelty Detection

Zhiwei Zhang, Yu Dong, Hanyu Peng, Shifeng Chen

Abstract—One-class novelty detection is conducted to identify anomalous instances, with different distributions from the expected normal instances. In this paper, the Generative Adversarial Network based on the Encoder-Decoder-Encoder scheme (EDE-GAN) achieves state-of-the-art performance. The two factors below serve the above purpose: 1) The EDE-GAN calculates the distance between two latent vectors as the anomaly score, which is unlike the previous methods by utilizing the reconstruction error between images. 2) The model obtains best results when the batch size is set to 1. To illustrate their superiority, we design a new GAN architecture (EDE-GAN), and compare performances according to different batch sizes. Moreover, with experimentation leads to discovery, our result implies there is also evidence of just how beneficial constraint on the latent space are when engaging in model training.

In an attempt to learn compact and fast models, we present a new technology, Progressive Knowledge Distillation with GANs (P-KDGAN), which connects two standard GANs through the designed distillation loss. Two-step progressive learning continuously augments the performance of student GANs with improved results over single-step approach. Our experimental results on CIFAR-10, MNIST, and FMNIST datasets illustrate that P-KDGAN improves the performance of the student GAN by 24.45:1, 311.11:1, and 700:1, respectively. The previous methods can be roughly divided into two categories based on using different anomaly score. The former methods use the residual loss between input and reconstructed images as anomaly score [44], [57], [43]. Fast AnoGAN [43] used the loss in the latent space for training, but did not use it as an anomaly score during inference. The above anomaly score based on pixel-to-pixel image matching is less robust to noises, which can be proved in our comparative experiments. The latter methods, such as Ganomaly [2] and Skip-Ganomaly [3], used the residual loss between two latent vectors (Figure 1(c)) on “MISO” tasks and achieved significant improvement. In this paper, we firstly utilize the EDE-GAN (Generative Adversarial Network based on Encoder-Decoder-Encoder architecture) model on “SIMO” task and achieves state-of-the-art performance when the batch size is set to 1. Previous works ignored an important factor, the impact of batch size in Batch Normalization (BN) [16] on the performance of one-class novelty detection. Different from traditional classification tasks, training data for one-class ND has only one category. In this paper, we would like to explain the influence of the batch size on one-class ND based on the following two points. 1) Model Fitting. As we know, BN is based on different channels to model all samples in a batch by calculating their mean and variance, assuming a normal distribution. If the batch size is larger than 1, the model’s ability to fit each sample will be weakened. 2) Generalization. The smaller batch size, meaning the larger gradient noise, impacts the generalization of model, and the results in Figure 1 demonstrate it well. In addition, we design a new GAN architecture (Figure 2(b), EDE-GAN) and perform comparative experiments to demonstrate the effectiveness of the constraint on the latent space.

However, deep neural networks with high computational costs and large storage prohibit their deployment to computation and memory resource limited systems. For tackling the above issue, neural network compression has been widely applied in recent years [8], [34], [35]. As one of the mainstream compression methods, Knowledge Distillation (KD) following a teacher-student paradigm transfers knowledge from a teacher network with higher performance to a student network. The early contributions used the outputs of

This manuscript is an extended version of the published paper, P-KDGAN: Progressive Knowledge Distillation with GANs for One-class Novelty Detection, IJCAI 2020, and is being submitted to IEEE Transactions on Image Processing for peer review, September 15, 2021.

Zhiwei Zhang, Yu Dong, Hanyu Peng, and Shifeng Chen are with Center for Multimedia Integrated Technologies, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China. Shifeng Chen is the corresponding author: shifeng.chen@siat.ac.cn.
In this paper, we apply EDE-GAN for one-class novelty detection that outperforms the state-of-the-art approaches. In order to deploy the deep neural networks in computation resources limited mobile devices, we propose the Progressive Knowledge Distillation with GANs (P-KDGAN) method to train the lightweight student network. The P-KDGAN approach improves the performance of student GAN by solving the following three problems. 1) How to design a distillation loss to measure the similarity of intermediate representations learned from the teacher GAN and the student GAN? The distillation loss $K_l$ described in Eq. 4 is designed as the weighted sum of the losses $K_1$, $K_g$ and $K_2$. 2) As shown in Figure 3 (b), how to combine the distillation loss $K_l$ with existing generator losses $L_g$, $L_d$ and discriminator losses $L_d$, $L_d^*$ from the student and teacher GANs to improve the performance of the student GAN? As is illustrated in Figure 3(b), we design four distillation structures based on different combinations of the above five losses. 3) Whether the designed four distillation structures can make the performance of the student GAN like that of the teacher GAN? If not, how to fix it? Our experimental results demonstrate that the performance of student GANs trained from scratch (or with random initialization) by the above four distillation structures is incomparable with the teacher GAN. Just as learning is a gradual cumulative process, a two-step progressive learning of KDGAN is proposed to continuously improve the performance of the student GAN. The detailed solutions will be illustrated in Section III.

The performance of proposed progressive knowledge distillation on GANs for one-class novelty detection is evaluated on CIFAR-10 [20], MNIST [21] and FMNIST [56] datasets. Our contributions are summarized as follows.

- We utilize the EDE-GAN for one-class novelty detection, which outperforms the state-of-the-art methods. We also design a new GAN architecture and perform comparative experiments to demonstrate the influence of the batch size and the effectiveness of the constraint on the latent space.
- We propose new distillation losses on latent vectors and reconstructed images of GANs to transfer knowledge from the teacher to student network. The distillation process is regarded as a knowledgeable teacher to improve the performance and stability of students through a two-step progressive learning, which includes basic knowledge learning and fine-learning.
- Progressive Knowledge Distillation with GANs is proposed for one-class novelty detection. Our experiments demonstrate that the P-KDGAN can improve the performance of the student GAN on the three datasets CIFAR-10, MNIST and FMNIST by 2.44%, 1.77%, and 1.73%, respectively.

## II. RELATED WORK

We briefly review the related works in term of one-class novelty detection and knowledge distillation.

### A. One-class Novelty Detection

In the semi-supervised one-class novelty detection, only the normal samples with one class are used for training the model. Conventionally, novelty detection methods can be
Fig. 2: Comparison with different knowledge distillation architectures.

Fig. 3: The flowchart of Knowledge Distillation with GANs for One-class Novelty Detection. (a) The Knowledge Distillation with Generative Adversarial Networks (KDGAN), in which the distillation losses $K_1$ ($K_1 = w_1 K_1 + w_2 K_x + w_2 K_2$) is designed for training the student GAN. (b) The two-step progressive learning of KDGAN is used to continuously improve the performance of the student GAN, in which P-KDGAN achieves the best performance. KDGAN-1, KDGAN-2, KDGAN-3 and KDGAN-4 are four different distillation structures.

divided into two categories. One is the traditional methods, such as One-Class SVM (OC-SVM) [45], Kernel Density Estimation (KDE) [32] and Principal Component Analysis (PCA) [53]. The disadvantage of such approaches is that they are not suitable for high-dimensional image data. The other methods based on deep learning include Deep Belief Networks (DBN) [10], Autoencoders (AE) [49] and generative adversarial networks (GANs) [44], [57], [36].

GANs have shown state-of-the-art performance in modeling complex high-dimensional image distributions [14]. Therefore, a lot of GANs based methods have been used for novelty detection [44], [57], [36]. The reconstruction error between inputs and reconstructed images or two latent vectors are utilized as the novelty score, which means that the learned model only reconstructs normal samples well, and shows very low tolerance for novel samples. Schlegl et al. [44] proposed the first GAN-based work, AnoGAN, for novelty detection. In training, the combination of the residual loss on images and the discrimination loss on feature maps is minimized to iteratively search the best latent vector. The Efficient GAN [57] based on BiGAN [9] network was proposed for jointly training the map from the image to the latent space simultaneously. Perera et al. [36] proposed the OCGAN in which two discriminators were used in the latent space and the input space for making the learned network better model the input images. Recently, Ganomaly [2] constructs a novel architecture EDE-GAN using the residual loss in latent space as anomaly score for “MISO” task. Skip-Ganomaly used the skip connections to thoroughly capture the multi-scale distribution of the normal data distribution in high-dimensional image space.

B. Knowledge Distillation

To reduce the large computation and storage cost of deep convolutional neural networks, knowledge distillation can transfer the generalization ability of a large network (or an ensemble of networks) to a light-weight network. Figure 2 (a), Hinton et al. [15] used the outputs of the softmax layer of a teacher network as the target function to train the student network. Romero et al. [39], illustrated in Figure 2.
(b), proposed that a student network with random initialization can imitate the intermediate representations of the teacher network to improve its own performance. In order to ensure the student network to learn the true data distribution from the teacher network, knowledge distillation with a discriminator was used for distinguishing features extracted from the teacher and student networks, Figure 2 (c) [51], [52], [25]. Li et al. [24] combined neural architecture search and knowledge distillation to compress the generator for controllable image synthesis.

GANs [14] have been applied to many real world applications such as domain adaptation, image generation, and anomaly detection. However, to our knowledge, there is no related works that deploy the knowledge distillation on two standard GANs. Therefore, this paper designs a distillation loss to transfer knowledge from the teacher GAN to the student GAN, in which knowledge distillation is considered as a progressive learning process that can continuously improve the performance of student networks.

III. METHOD

In this section, we firstly analyze the impacts of the batch size on one-class novelty detection, and then introduce three different network architectures that will be used to perform comparative experiments to illustrate the superiority of EDE-GAN. Afterwards, the distillation loss based on the EDE-GAN model and four different distillation structures are proposed to transfer knowledge. Finally, we propose a two-step Progressive Knowledge Distillation with GANs (P-KDGAN) to learn a light-weight student GAN from a pre-trained teacher GAN.

A. Analysis on Batch Size

The evaluation metric, Area Under Curve (AUC), can be written as:

\[
AUC = \frac{\sum_{i\in Normal} R_i - N \times (N+1)}{M \times N} \tag{1}
\]

where \(R\) denotes the index of sorted prediction probabilities, and \(M, N\) represent the number of samples on the normal and abnormal classes.

Based on the Eq. 1 in order to improve the generalization of model, we need to reduce the over-fitting on normal training data, i.e. reduce their prediction confidences to increase the sum of the index values of normal samples \(\sum_{i\in Normal} R_i\). As we know, for different batch sizes, there are trade-offs between fitting and generalization. When the batch size equals to 1, as illustrated in Figure 4 and Figure 5 the models can achieve the best performances. The above results demonstrate that the models can well learn the identity of each normal sample, at the same time, large gradient noise and the constraint on the latent space avoid the over-fitting. However, when the batch size is larger than 1, we can think that the fitting ability of the model is greatly weakened. Because the BN needs to model features of different samples on the channels, i.e., a single sample will be severely affected by the same min-batch samples, resulting in poor testing performance and difficulty in optimization [55].

B. Three GAN Architectures

GAN-based novelty detection emerges quickly as one of the popular detection approaches after its early use in [44]. GAN consists of two adversial modules, a generator \(G\) that learns the distribution of the input image \(\mathbf{x}\) from the latent spaces, and a discriminator \(D\) that decides whether the reconstructed image \(\hat{\mathbf{x}}\) is real or fake. \(D\) and \(G\) are simultaneously optimized by playing the following minimax game as:

\[
\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim \mathbf{X}} [\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim \mathcal{Z}} [\log (1 - G(\mathbf{z}))], \tag{2}
\]

where training dataset \(\mathbf{X}\) comprises \(N\) normal images, \(\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N] \in \mathbb{R}^{N \times w \times h \times c}\), and \(\mathbb{E}_{\mathbf{x}\sim \mathbf{X}}\) is the expected value of \(\mathbf{x}\) obeying the distribution of normal images \(\mathbf{X}\).

Figure 4 shows three different generator architectures for novelty detection, in which the discriminators are not used during inference. The “dotted” lines indicate that the modules do not participate in the inference progress. \(L_x\) is the reconstruction error between two latent vectors \(z_1\) and \(z_2\). (c) \(G_{EDE}\) based on encoder-decoder-encoder scheme uses the \(L_z\) as the anomaly score.

Fig. 4: Comparison with different generator architectures for novelty detection, in which the discriminators are not used during inference. (a) \(G_{ED}\) based on encoder-decoder pipeline uses the reconstruction error \(L_x\) between images as the anomaly score. (b) \(G_{ED,E}\) is used for training where \(G_E\) and \(L_z\) do not participate in the inference progress.
where \( f \) of the images, latent vectors and feature maps, respectively.

In order to demonstrate the superiority of \( G_{ED} \) shown in Figure 4 (a) uses the reconstruction error \( L_x \) (\( L_x = \| x - \hat{x} \|_1 \)) between input image \( x \) and reconstructed image \( \hat{x} \) as the anomaly score for detection, and there is no constraint on the latent space.

\( G_{EDE} \) based on the encoder-decoder-encoder pipeline, illustrated in Figure 4(c), uses the residual loss \( L_x \) (\( L_x = \| z - \hat{z} \|_2 \)) as the anomaly score, and the constraint \( L_x \) is used for training.

In order to demonstrate the superiority of \( G_{EDE} \) and the effectiveness of the constraint on the latent space, we design a new GAN architecture \( G_{EDE} \) (Figure 4 (b)), in which the second encoder \( G_e \) and constraint loss \( L_x \) are only used during training, hence \( L_x \) is the novelty score.

### C. KDGAN

In the KDGAN, we design a distillation loss based on EDEGAN to solve problem (1) described in Section II. As shown in Figure 3, the difference between them is the number of channels in each layer. The generator loss \( L_g^S \) of the student GAN includes reconstructed image loss \( S_{con} \), latent space loss \( S_{enc} \) and adversarial loss \( S_{adv} \):

\[
S_{con} = \mathbb{E}_{x \sim \mathcal{X}} \| x - \hat{x} \|_1, \tag{3}
\]

\[
S_{enc} = \mathbb{E}_{x \sim \mathcal{X}} \| z_1 - \hat{z}_2 \|_2, \tag{4}
\]

\[
S_{adv} = \mathbb{E}_{x \sim \mathcal{X}} \| f(x) - f(\hat{x}) \|_2, \tag{5}
\]

\[
L_g^S = w_{con} S_{con} + w_{enc} S_{enc} + w_{adv} S_{adv}, \tag{6}
\]

where \( f(\cdot) \) outputs the intermediate representations of discriminator \( D \). \( S_{con}, S_{enc} \) and \( S_{adv} \) denote the reconstruction errors of the images, latent vectors and feature maps, respectively. The weighted sum of \( S_{con}, S_{enc} \) and \( S_{adv} \) constitutes the generator loss \( L_g^S \) and is minimized to train the student generator \( G \).

The designed distillation loss \( K_i \) is a novel attempt for knowledge distillation on two standard GANs. As shown in Figure 3 (a), the teacher GAN and the student GAN transfer knowledge through the intermediate layers of the generators, which includes two latent vectors and one reconstructed images. In the KDGAN, we design three losses \( K_1, K_x \) and \( K_2 \) to measure the similarity of the intermediate layers. \( K_1 \) and \( K_2 \) are the \( L_2 \) distance of latent vectors (\( z_1, z_2, x_1 \) and \( x_2 \)) from the teacher GAN and student GAN. \( K_x \) is the \( L_1 \) distance of reconstructed images (\( \hat{x}, \hat{x} \)). Based on the above three losses, we propose distillation loss \( K_i \) as an objective function for knowledge distillation which is the weighted sum of \( K_1, K_x \) and \( K_2 \):

\[
K_i = w_1 K_1 + w_2 K_x + w_2 K_2, \tag{7}
\]

\[
K_1 = \| z_1 - \hat{z}_1 \|_2, \tag{8}
\]

\[
K_2 = \| z_2 - \hat{z}_2 \|_2. \tag{9}
\]

We summarize the symbols and their corresponding equations in Table II.

### D. Four Distillation Structures

As is illustrated in Figure 3(a), the designed distillation loss \( K_i \) builds a “bridge” between the teacher and student GANs for knowledge transfer. The losses in the KDGAN consist of three parts: teacher GAN losses \( L_{T}^T \), student GAN losses \( L_{S}^S \) and distillation loss \( K_i \). We define the above five loss functions as the elements of set \( L \):

\[
L = \{ \alpha L_{T}^T, \beta L_{T}^T, \mu L_{S}^S, \nu L_{S}^S, \lambda K_i \}, \tag{9}
\]

where \( \alpha, \beta, \mu, \nu, \) and \( \lambda \in \{0,1\} \) indicates whether the corresponding loss is used to train the networks.
The elements in $\mathcal{L}$ can be combined into four subsets ($\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{L}_3$, $\mathcal{L}_4$) to form different distillation structures according to the following two rules. The first rule is whether the teacher GAN has fixed weights; the second rule is whether the distillation loss $K_t$ is combined with the losses $L_g^S$, $L_d^S$ to train the student GAN. Before the KDGAN, a teacher GAN is trained by its own generator loss $L_g^T$ and discriminator loss $L_d^T$. The designed four distillation structures are introduced as follows.

- **KDGAN-1**: $\mathcal{L}_1=\{K_t\}$. Without the use of real labels, the training of the student network only depends on the distillation loss $K_t$, which results in poor detection performance. There is no adversarial networks, and the teacher network is not updated, so its training speed is the fastest.

- **KDGAN-2**: $\mathcal{L}_2=\{L_g^S, L_d^S, K_t\}$. The student GAN is trained by minimizing its own losses $L_g^S$, $L_d^S$ and distillation loss $K_t$, while the teacher GAN is not updated. The adversarial network in student GAN causes its training speed to be slightly slower than KDGAN-1.

- **KDGAN-3**: $\mathcal{L}_3=\{L_g^T, L_d^T, K_1\}$. The teacher GAN uses its own losses $L_g^T$, $L_d^T$ to train to maintain its performance, when the training of the student GAN follows KDGAN-1. Its training speed is almost the same as that of KDGAN-2.

- **KDGAN-4**: $\mathcal{L}_4=\{L_g^T, L_d^T, L_g^S, L_d^S, K_1\}$. The trainings of the teacher and student GANs follow KDGAN-3 and KDGAN-2, respectively. There are two adversarial networks that need to be trained simultaneously, so the training speed is the lowest.

**E. Progressive Learning of KDGAN**

The progressive learning of KDGAN, shown in Figure 3 (b), is a two-step approach that continuously improves the performance of the student GAN and achieves better performance than those single step methods. The two-step P-KDGAN is described as follows.

1) **P-KDGAN-I**: In the first step, four distillation structures are utilized to train the student network. The experimental results shown in Section IV-E demonstrate that the performance of the student network with random initialization has a large gap compared with the teacher network. Therefore, considering the detection accuracy and training time of the four distillation structures, KDGAN-2 is used as the first step of P-KDGAN to enable the student network to learn the basics knowledge from the teacher network. In KDGAN-2, the pre-trained teacher has already converged, so the teacher network with fixed weights is used to train the student network relying on real labels and distillation knowledge. Such a “teaching by teacher” step makes the student learn the basic knowledge totally from the teacher.

2) **P-KDGAN-II**: In the second step, KDGAN-3 and KDGAN-4 continue to train the teacher networks, while the student networks with basic knowledge rely on distilling knowledge to fine-training, thereby further improving accuracy and stability. The fine-learning processes in this step are named as P-KDGAN-II-3 and P-KDGAN-II-4. The experimental results prove that the performance of the student network even exceeds the teacher network in some categories of one-class novelty detection.

The above process is illustrated in Algorithm 1.

**Algorithm 1 Progressive Knowledge Distillation with GANs**

**Input:** Pre-trained teacher $G$ and $D$, training dataset with normal instances $(x_i, y_i)_{i=1}^N$, epoch $T$

**Output:** Improved student $G$

1. **First step**
2. Student $G$ and $D$ with random initialization
3. for $t=1$ to $T$ do
4. for $i=1$ to $N$ do
5. Update student $G$ and $D$ when teacher GAN with fixed weights
6. end for
7. end for

8. **Second step**
9. Download the weights of the teacher and student GANs from the previous last epoch
10. for $t=1$ to $T$ do
11. for $i=1$ to $N$ do
12. Update teacher $G$ and $D$
13. Update student $G$ and $D$
14. end for
15. end for

In this section, the proposed P-KDGAN is evaluated on the well-known CIFAR-10 [20], MNIST [21] and FMNIST [56] datasets. Following previous work [36], we quantify the performance of our method using the Area Under Curve (AUC) of Receiver Operating Characteristics (ROC). The detailed analysis on the results and comparative experiments with state of the art techniques are introduced as follows.

All the reported results are implemented using the PyTorch framework [33] on NVIDIA RTX 2080TI. In our experiments, the batch size and epoch are set to 1 and 500 respectively. We apply Adam [18] with $\beta_1 = 0.5$ and $\beta_2 = 0.99$ to optimize model parameters with a learning rate of 0.002.

**A. Datasets**

For the three experimental datasets, the training and testing partitions remain as default. In the setup, one of the classes from training dataset is considered as normal samples for training. During testing, the remaining classes are used to represent novelty samples. For example, every experiment on the CIFAR-10 dataset is trained with 5000 samples and tested with 10,000 samples. The above experiments are repeated for all ten categories. In addition, in order to be compatible with the network architectures, all images are resized to $32 \times 32$ by Bilinear interpolation.

- **CIFAR-10**: consists of 60,000 $32 \times 32$ RGB images in 10 classes, with 6,000 images for per class. Each category contains 50,000 training images and 10,000 test images.
B. Network Architectures

The encoder $E(x)$ and decoder $D(x)$ in ED-GAN, ED$\hat{E}$-GAN and EDE-GAN frameworks follow the DCGAN [38] architecture, which have three basic layers in our model. As shown in Table II, the basic layers consist of convolutional layers (deconvolutional layers), batch normalization and activation. In contrast, LeakyReLU and ReLU activations are used in encoders and decoders, except for the last layer in decoder, which uses Tanh. All the convolution filters are set to $4 \times 4$.

In the following P-KDGAN experiments, the difference between a teacher network and a student network is the number of channels in the intermediate representations. For the three experimental datasets, the intermediate layers in the teacher networks are set to 64-128-256 channels following the OCGAN [36]. The student networks in each dataset utilize intermediate representations with 8-16-64 channels, 2-4-8 channels and 1-2-4 channels respectively ($u = 8, 2, 1$ respectively). The encoder $E(x)$ and decoder $D(x)$ architecture of the teacher GAN is illustrated in Table II.

C. Comparison with EDE-GAN

As shown in Figure 5 the EDE-GAN model achieves the best results in the “Last” performance on three datasets.

### Table II: The encoder and decoder architectures for our teacher GAN and student GAN, layer by layer. Units refer to number of filters in the case of convolution layers, and BN is Batch Normalization abbreviated. $u$ represents the number of channels in the layer, and it has different value on every dataset, that is, student network with different size of parameters.

| Layer | Teacher-Units | Student-Units | BN | Activation | Kernel |
|-------|---------------|---------------|----|------------|--------|
| $E(x)$ |               |               |    |            |        |
| Conv2D | 64           | $u$           | ✓  | LeakyReLU  | $4 \times 4$ |
| Conv2D | 128          | $2 \times u$  | ✓  | LeakyReLU  | $4 \times 4$ |
| Conv2D | 256          | $4 \times u$  | ✓  | LeakyReLU  | $4 \times 4$ |
| Conv2D | 256          | 256           | ✓  |            | $4 \times 4$ |
| $D(x)$ |               |               |    |            |        |
| ConvTrans2D | 256 | $4 \times u$  | ✓  | ReLU       | $4 \times 4$ |
| ConvTrans2D | 128 | $2 \times u$  | ✓  | ReLU       | $4 \times 4$ |
| ConvTrans2D | 64  | $u$           | ✓  | ReLU       | $4 \times 4$ |
| ConvTrans2D | 3   | 3             |    | Tanh       | $4 \times 4$ |

### Table III: The hyper-parameters in our experiments.

| Hyper-parameter | Value |
|-----------------|-------|
| $w_{con}$       | 10    |
| $w_{enc}$       | 1     |
| $w_{adv}$       | 1     |
| $w_1$           | 1     |
| $w_x$           | 1     |
| $w_2$           | 1     |

### MNIST

- **MNIST**: consists of 70,000 $28 \times 28$ hand-written grayscale digit images from 0-9. The training set has 60,000 images and the test set has 10,000 images.

### FMNIST

- **FMNIST**: similars to MNIST, in which the images are from an online clothing store. The $28 \times 28$ grayscale images of 70,000 fashion products from 10 categories, with 7,000 images per category. The number of training and testing images are as same as MNIST.
| Dataset  | Method     | AUC. ↓ | #Param. ↓ | #FLOPs. ↓ |
|----------|------------|--------|-----------|-----------|
| CIFAR-10 | Teacher    | 73.76% | 5.12M     | 56M       |
|          | Student    | 3.15%  | 6.22×     | 24.45×    |
|          | P-KDGAN    | 0.71%  |           |           |
| MNIST    | Teacher    | 97.80% | 5.12M     | 56M       |
|          | Student    | 2.32%  | 52.22×    | 311.11×   |
|          | P-KDGAN    | 0.55%  |           |           |
| FMNIST   | Teacher    | 93.11% | 5.12M     | 56M       |
|          | Student    | 1.91%  | 105.45×   | 700×      |
|          | P-KDGAN    | 0.18%  |           |           |

TABLE V: Evaluation of our P-KDGAN method on CIFAR-10, MNIST and FMNIST datasets. (M means million, # means the compression ratio of parameter numbers and FLOPs compared to the teacher GAN.)
and exceeds the runner-up by 5.74%, 36.92% and 27.15% respectively excepting for the “BEST” performance on the CIFAR-10 data. Compared with ED-GAN, the new designed ED{E}-GAN improves the “LAST” performance by 0.91%, 4.66% and 2.08% on three datasets respectively by using the constraint on the latent vectors for training when \( B = 1 \). The above experimental results illustrate the beneficial constraint on the latent space, and the novelty score on latent vectors is more robust to noise.

D. Results on One-class Novelty Detection

In this section, we compare our Ganomaly [2] based Teacher GAN with several traditional and deep learning based methods on CIFAR-10 and MNIST datasets, including one-class SVM (OC-SVM) [43], kernel density estimation (KDE) [32], deep variational autoencoder (VAE) [19], AND [1], AnoGAN [44], DSVDD [40] and OCGAN [36]. In light of massive experiments, the parameters of \( w_{\text{con}}, w_{\text{enc}} \) and \( w_{\text{adv}} \) in Eq. [6] are manually configured as 10, 1 and 1. The parameters of \( w_1, w_x \) and \( w_2 \) in Eq. [7] are set as 1. We take the average AUC of the last epoch from multiple trials, but not the manually selected result, as the detection performance, which is more convicive.

**Comparisons on CIFAR-10 and MNIST.** For one-class novelty detection on CIFAR-10 dataset, our method achieves the result of 73.76% as shown in Table [IV] which is higher than the best OCGAN [36] method about 8%. For MNIST dataset, our method achieves 97.80% yielding an improvement of about 0.3% compared with the state-of-the-art method.

E. Evaluation of P-KDGAN Method

In this section, the progressive knowledge distillation with GANs is evaluated on CIFAR-10, MNIST and FMNIST datasets, which illustrates the effectiveness of our progressive learning of KDGAN. Although KDGAN-\( \{3 \} \) achieves the second-best results on MNIST and FMNIST, it shows the worst performance on CIFAR-10 dataset. KDGAN-\( \{3 \} \) obtains...
the second-best results on CIFAR-10, but it is about 0.5% lower than the best result. In addition, KDGAN-d (KDGAN-④) illustrated in Figure[4]c) is inferior in accuracy and training stability compared to P-KDGAN-II. The training curves of the AUC illustrated in Figure[3] clearly show that proposed P-KDGAN-II can improve the accuracy of the student network and even surpass the teacher network, and reduce shock. Therefore, the above analysis concludes that student networks with random initialization can only learn the basic knowledge of the teacher networks, and the fine-training in the second step of P-KDGAN can further improve performance.

**Results on P-KDGAN.** As illustrated in Table [V], the performance of the student GAN obtained by two-step P-KDGAN is only 0.71%, 0.55% and 0.18% lower than that of the teacher GAN when compressing the computation at ratios of 24.45:1, 311:1:1, and 700:1, respectively.

V. CONCLUSION AND DISCUSSION

In this paper, we use the EDE-GAN for one-class novelty detection and achieve state-of-the-art performance. We firstly design a new GAN architecture and perform comparative experiments to demonstrate the beneficial constraint on the latent space. To compress the model, the progressive knowledge distillation with GANs is proposed, which is a novel exploration that applies the knowledge distillation on two standard GANs. The two-step progressive learning can continuously improve the performance and reduce shock of the student network, in which the designed distillation loss plays an important role. Experiments on three datasets validate the effectiveness of our proposed method.

We hope the proposed method will open new avenues of solution for one-class novelty detection and other related problems, and make the following summary:

1) Re-evaluate the performance of all other methods on the one-class novelty detection task based on different batch sizes, and we need to consider the performance of model in the last epoch, rather than selecting the best result in all epochs for display.

2) We are looking forward to the application of our proposed P-KDGAN on other tasks, such as image dehazing, image super-resolution and image synthesis.

3) Since novelty detection is different from traditional classification problem, we may need to rethink how Batch Normalization can be improved on this task.

VI. ACKNOWLEDGMENT

This work is supported by Key-Area Research and Development Program of Guangdong Province (2019B010155003), Shenzhen Science and Technology Innovation Commission (JCYJ2020101519011208), and National Natural Science Foundation of China (U1713203).

REFERENCES

[1] Davide Abati, Angelo Porrello, Simone Calderara, and R. Cucchiara. Latent space autoregression for novelty detection. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 481–490, 2019.

[2] Samet Akcay, Amir Atapour-Abarghouei, and Toby P. Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In Asian conference on computer vision, pages 622–637. Springer, 2018.

[3] S. Akcay, Amir Atapour-Abarghouei, and T. Breckon. Skip-ganomaly: Skip connected and adversarially trained encoder-decoder anomaly detection. In 2019 International Joint Conference on Neural Networks (IJCNN), pages 1–8, 2019.

[4] Philippe Burlina, Neil Joshi, and J-Jeng Wang. Where’s wally now? deep generative and discriminative embeddings for novelty detection. CVPR, pages 11499–11508, 2019.

[5] Raghavendra Chapalapathy, Aditya Krishna Menon, and Sanjay Chawla. Anomaly detection using one-class neural networks. arXiv preprint arXiv:1802.06360, 2018.

[6] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection : A survey. ACM Comput. Surv., 41(3):1–72, 2009.

[7] Jiacheng Cheng and Nuno Vasconcelos. Learning deep classifiers consistent with fine-grained novelty detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1664–1673, 2021.

[8] Y. Cheng, D. Wang, P. Zhou, and T. Zhang. A survey of model compression and acceleration for deep neural networks. ArXiv, abs/1710.09282, 2017.

[9] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. ICLR, 2017.

[10] Sara 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.

[11] Pedro García-Teodoro, Jesús E. Díaz-Verdejo, Gabriel Maciá-Fernández, and Enrique Vázquez. Anomaly-based network intrusion detection: Techniques, systems and challenges. Comput. Secur., 28(1-2):18–28, 2009.

[12] Izhak Golan and Ran El-Yaniv. Deep anomaly detection using geometric transformations. In Advances in Neural Information Processing Systems, volume 31, pages 9758–9769, 2018.

[13] Dong Gong, L. Liu, Vuong Le, Budhadipta Saha, M. Mansour, S. Venkatesh, and A. V. D. Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pages 1705–1714, 2019.

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

[15] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. ArXiv, abs/1503.02531, 2015.

[16] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International conference on machine learning, pages 448–456. PMLR, 2015.

[17] Eric Jardim, Lucas A Thomaz, Eduardo AB da Silva, and Sergio L Netto. Domain-transformable sparse representation for anomaly detection in moving-camera videos. *IEEE Transactions on Image Processing*, 29:1329–1343, 2019.

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

[19] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *ICLR*, pages 1705–1714, 2019.

[20] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[21] Yann LeCun and Corinna Cortes. The mnist database of handwritten digits. 2005.

[22] Roberto Leyva, Victor Sanchez, and Chang-Tsun Li. Video anomaly detection with compact feature sets for online performance. *IEEE Transactions on Image Processing*, 26(7):3463–3478, 2017.

[23] Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9664–9674, 2021.

[24] Mingyang Li, Ji Lin, Yaoyao Ding, Zhijian Liu, Jun-Yan Zhu, and Song Han. Gan compression: Efficient architectures for interactive conditional gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5284–5294, 2020.

[25] Peiye Liu, Wu Liu, Huadong Ma, Tao Mei, and Mingoo Seok. Ktan: knowledge transfer adversarial network. ArXiv, abs/1810.08126, 2018.

[26] Minjie Liu, Yu Cheng, Cao Xiao, Shiyu Chang, Shuai Huang, Bin Liang, and Thomas Huang. Unsupervised sequential outlier detection with deep architectures. *IEEE transactions on image processing*, 26(9):4321–4330, 2017.
