A Hierarchical Transformation-Discriminating Generative Model for Few Shot Anomaly Detection

Shelly Sheynin[1*] Sagie Benaim[1*] Lior Wolf[1,2]
[1]The School of Computer Science, Tel Aviv University
[2]Facebook AI Research

Abstract

Anomaly detection, the task of identifying unusual samples in data, often relies on a large set of training samples. In this work, we consider the setting of few-shot anomaly detection in images, where only a few images are given at training. We devise a hierarchical generative model that captures the multi-scale patch distribution of each training image. We further enhance the representation of our model by using image transformations and optimize scale-specific patch-discriminators to distinguish between real and fake patches of the image, as well as between different transformations applied to those patches. The anomaly score is obtained by aggregating the patch-based votes of the correct transformation across scales and image regions. We demonstrate the superiority of our method on both the one-shot and few-shot settings, on the datasets of Paris, CIFAR10, MNIST and FashionMNIST as well as in the setting of defect detection on MVTec. In all cases, our method outperforms the recent baseline methods.

1. Introduction

Anomaly detection [5, 1] is the task of detecting unusual samples in the data. In the typical setting of *one class classification* [20], given a large collection of samples from the *normal* (non-anomalous) data, the learner is asked to classify novel samples as either normal or anomalous. In this paper, we aim to solve this problem given very few training samples, including the case of a single training sample. Our study is motivated by the scarcity of training samples in many visual domains, as well as by the human ability to solve this task after observing a very limited number of samples [7, 8].

Our model relies on two main components. The first component is a hierarchical generative model that captures the internal patch statistics of one or few images at multiple scales. This component follows the recent successes of deep generative models to generate multiple, varied, natural looking images, given a single training image [28, 12, 35]. We generalize this setting to few images, by adding conditioning on the image index, thus enabling information sharing between multiple training images.

Modeling patches at different scales, allows the detection of anomalies both in global properties, such as color or large-scale structural changes, and in local regions. The multi-scale patch based approach is also useful when considering the task of defect detection, where the anomaly may only manifest in few well localized regions.

The second component is a *self-supervised learning* task. Recent approaches [10, 3, 11] have shown that, in the many-shot case, a classifier trained on a proxy task, such as identifying the transformation applied to the input, accurately captures novel samples that are similar to the training data, and thus can distinguish between normal and anomalous samples. We utilize such a proxy task in the context of our multi-scale generative model. For each scale, the generated patches are transformed according to a predefined set of transformations. A discriminator is asked to distinguish between real and fake samples, as well as between the transformations performed.

Our method builds these components into a single model. At test time, a sample is considered anomalous, if many of its patches, at all scales, are determined anomalous, according to the learned patch-specific discriminators. In a comprehensive set of the experiments, we demonstrate that the proposed method outperforms recent baselines on anomaly detection benchmarks. This is true for the one-shot, five-shot and ten-shot settings. When the number of training samples increases, our method is shown to improve in performance. In the field of defect detection, we show that our method outperforms, in the few shot setting, the state of the art in localizing the anomalous patch.

2. Related Work

Image based anomaly detection Image based anomaly detection methods can be divided into the categories of *reconstruction*, *classification* and *distribution* based methods [30].

*Equal contribution

Reconstruction based methods represent data in a manner whereby normal data can be reconstructed with small error, while anomalous data incurs a high reconstruction error.

Classification based methods attempt to discriminate between data regions of normal data and those of anomalous data. Ruff et al. [31] proposed DeepSVDD, a deep learning variant of SVDD, which maps images to a more meaningful deep feature space. PatchSVDD [39] improved this method, by extending it to a patch-based method using self-supervised learning. Recent *self-supervised* approaches attempt to find a "proxy" classification objective, such that classifying normal data based on this objective, allows for a good separation of normal and anomalous data. For instance, Goyal et al. [11] trains a classifier to distinguish training samples from their perturbations generated adversarially. Golan and El-Yaniv [10] use a set of predefined transformations. They train a classifier to distinguish between the type of transformation performed on normal data and show that such a classifier can be used to distinguish between normal and anomalous data. While our method uses such a proxy objective, it is used within an hierarchical patch based generative setting. This allows the modeling of the patch based distribution of images together with a discriminative capability to model the regions of this distribution.

Distribution based methods model the distribution of normal data. Deep generative models have shown great promise in modeling complicated distributions. In particular, autoencoders and variational autoencoders [40, 2], as well as GAN based approaches [33] are typically used. Nalisnick et al., [21] however, argue that a generative model alone may not be sufficient to detect out-of-distribution inputs.

Unlike these approaches, our model benefits from the use of both *distribution* and *classification* based components: (1) A hierarchical generative model used to model the internal multi-scale patch distribution of a single or few images and (2) A scale-dependent discriminator, which learns to distinguish between real and fake image patches, as well as between transformations of such patches. Our model incorporates these components into a single model. Unlike previous approaches, this allows for the detection of anomalous samples in the case where, during training, only a single or a few images are given from the normal class.

Few shot learning The use of limited supervision for image classification has been studied extensively [37, 6, 36]. Our work relates more closely to anomaly detection works that use limited supervision. Some works [22, 24] consider the setting where a limited number of samples is given from the anomalous classes, but many samples are given from the normal class. Kozerawski et al., [15] use large labelled dataset generated from ImageNet and apply transfer learning. Other works [23, 25] tackle a slightly different problem, where few samples are given from novel anomalous classes. Frikha et al., [9] and Kruspe [17] consider a meta-learning approach where few samples are given from many classes at training. All of these works use additional supervision, not used in our method. Our work assumes uses only a limited number of samples from the normal class.

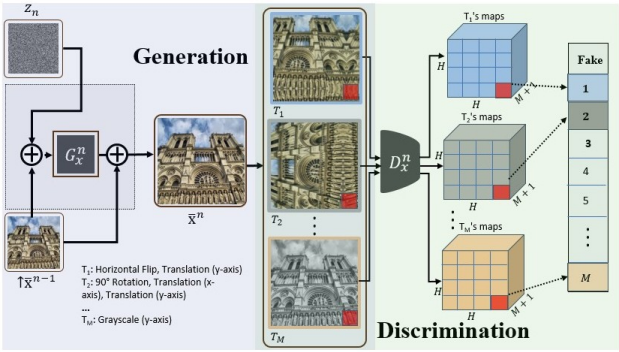


Figure 1. **Training pipeline at scale n for the one shot setting.** Generator G_x^n receives as input, the upsampled image from previous scale $\overline{x}^{n-1}$ and the noise map z^n, and generates a new sample $\overline{x}^n$. $\overline{x}^n$ is transformed into the set $\{T_1(\overline{x}^n), \ldots, T_M(\overline{x}^n)\}$ of M transformed images and fed into the multi-class discriminator D_x^n. For each T_i, the result is $M+1$ maps of size $H \times H$. Each $M+1$ sized vector of this $H \times H$ map represents a softmax probability vector over $M+1$ classes for patch p (for instance the red patch at the bottom right). For each such vector, G_n^x attempts to maximize the probability of class i for an image transformed using T_i.

Learning the Internal Statistics Our work is related to recent works that model the internal distribution of images [42, 35, 28, 12]. These works present a GAN capable of generating from the internal patch distribution of a single image. However, these methods are used in the context of generation, and are designed on a single image only.

3. Method

Let $\mathbb{X}$ be the set of all natural images and let $X \subset \mathbb{X}$ be the subset of images in the class that is defined by the training images, i.e., the normal class. Images residing outside X are considered anomalous. We are given an i.i.d randomly selected subset of images $X_k = \{x_1, \ldots, x_k\} \subset X$. The task is to learn a classifier C_{X_k} from the training set X_k, such that $C_{X_k}(x) = 1$ if $x \in X$ and $C_{X_k}(x) = 0$ otherwise. We are interested in the case where k is small.

3.1. One Shot Anomaly Detection

We begin with the case of $k = 1$, where the training set contains a single sample x_1, denoted as x for brevity. While x alone cannot capture the intra-class variability of the normal class, we argue that the image itself contains informative structure that can help identify the class. An illustration of our training pipeline is provided in Fig. 1.

Internal distribution of patches Our sample space can be significantly enriched by considering different image

patches, at different scales. To this end, we model the distribution of the patches of x at different scales, using a similar pipeline to SinGAN [28]. Unlike SinGAN, we use a multi-class discriminator to distinguish between different classes of transformations applied on real and generated samples.

For a given scale $n = 0, 1, 2, \ldots, N$, we denote by p_x^n the distribution of images that have the same patch distribution at this scale as x. In other words, x is composed of many patches at a given distribution and we model the distribution p_x^n of images that have the same scale-dependent patch distribution. For each scale n, the downsampling factor is r^{N-n} for some $r > 1$. Scale 0 is of the lowest resolution and scale N is of the highest (original) image resolution. A bicubic downsampling is used.

At each scale n, our method employs a patch GAN [43, 19] for generating samples in p_x^n. Each patch-GAN consists of a fully convolutional generator G_x^n and discriminator D_x^n with a receptive field of size 11×11. The architecture of G_x^n and D_x^n and training details are outlined below in Sec. 3.4. Let x^n be x, down-sampled by a factor of r^{N-n}, and z^n be Gaussian noise of the same dimension and shape as x^n. At the coarsest scale, $n = 0$, the output of G_x^n is defined as (the overline is used to denote generated images):

$$\overline{x}^0 = G_x^0(z^0) \tag{1}$$

Moving to finer scales $n > 0$, G_x^n accepts as input both z^n and an upscaled image of $\overline{x}^{n-1}$ to the dimension of x^n, denoted by $\uparrow \overline{x}^{n-1}$. A residual generation is then used:

$$\overline{x}^n = G_x^n(z^n + \uparrow \overline{x}^{n-1}) + \uparrow \overline{x}^{n-1}\,. \tag{2}$$

This way, the network G_x^n adds the missing details of $\uparrow \overline{x}^{n-1}$ that are specific to scale n.

Transformations To enhance the ability of our model to represent p_x^n and inspired by recent work [41, 13] on few shot generation, we apply a fixed set of $\mathcal{M}$ differentiable transformations, $T_1, \ldots, T_M$, to all real and generated images given as input to D_x^n. These transformations are chosen, such that, on one hand, they enrich the sample space captured by the current model, thus enabling it to more faithfully capture p_x^n, while on the other hand, they do not produce samples outside the class of x.

We use the set of transformations which result from applying the following transformations sequentially: (1) horizontal flip, (2) translation: the image is shifted by a ratio of 0.15 in x axis, y axis or both, (3) 90° rotations $\{\mathcal{R}_0, \mathcal{R}_{90}, \mathcal{R}_{180}, \mathcal{R}_{270}\}$, and (4) color transformation of RGB to gray-scale. For grayscale image datasets, we do not use color transformations (4). Due to memory constraints, we use a subset of $M = 54$ (42 for grayscale image datasets) transformations from this group, where the same set of transformations are used for all datasets and experiments. The exact set of transformations is given in the supplementary. We avoid undesirable effects at image borders, by first padding the image with reflection. As a pre-processing step, before applying the transformations, we apply histogram equalization to all the training images in RGB color space.

Training objectives $T_1, \ldots, T_M$ are used in training G_x^n and D_x^n. For each scale, we transform x^n into the set $\{T_1(x^n), \ldots, T_M(x^n)\}$ of M transformed real images. Similarly, each generated sample $\overline{x}^n$ is transformed into $\{T_1(\overline{x}^n), \ldots, T_M(\overline{x}^n)\}$. Additionally, the task of recognizing the underlying transformation enriches the output space of D_x^n, in a way that is suitable for anomaly detection.

D_x^n is a fully convolutional Markovian discriminator [43, 19] of the same architecture as G_x^n except that the number of output channels in the last convolutional layer is $\mathcal{M} + 1$. D_x^n is trained in a discriminative fashion to classify all the patches of $T_i(x^n)$ as i. In addition, it is optimized to classify all the patches of $T_i(\overline{x}^n)$ to the "fake" class 0. G_x^n tries to fool D_x^n by producing $\overline{x}^n$ such that all the patches of $T_i(\overline{x}^n)$ are classified as i. For each T_i, D_x^n produces $M + 1$ maps of size $H \times H$. Applying softmax, along the $M + 1$ dimensions, produces pseudo-probabilities for patch p of the input to belong to one of the $M + 1$ classes. We use the following adversarial loss terms:

$$\mathcal{L}_{adv}^D(D_x^n) = \sum_{i=1}^{M} \sum_{p \in H \times H} \mathcal{L}_{CE}(D_x^n(T_i(x^n))_p, i) \tag{3}$$

$$-\mathcal{L}_{CE}(D_x^n(T_i(\overline{x}^n))_p, 0) \tag{4}$$

$$\mathcal{L}_{adv}^G(G_x^n) = \sum_{i=1}^{M} \sum_{p \in H \times H} \mathcal{L}_{CE}(D_x^n(T_i(\overline{x}^n))_p, i) \tag{5}$$

$$\mathcal{L}_{adv}(D_x^n, G_x^n) = \mathcal{L}_{adv}^G(G_x^n) - \mathcal{L}_{adv}^D(D_x^n) \tag{6}$$

where $D_x^n(T_i(x^n))_p$ (similarly $D_x^n(T_i(\overline{x}^n))_p$) denotes the softmax probability vector of size $M + 1$ for point p in the $H \times H$ map produced by D_x^n and $\mathcal{L}_{CE}$ denotes the cross entropy loss. Recent literature [10] has shown the potential of training a classifier on transformation detection for anomaly detection. We utilize a similar discriminative objective, but in the context of a multi-scale hierarchical generative model. As shown in Sec. 4, the use of this hierarchical generative mode significantly improves results.

In addition to adversarial training, a reconstruction loss is used. For $n = 0$, G_x^0 attempts to reconstruct x_0 given a fixed random noise z^* while for $n > 0$, G_x^n attempts to reconstruct x^n given an upsampled version of $\overline{x}^{n-1}$ that is obtained, recursively, based on z^*, without further randomness:

$$\overline{\overline{x}}^0 = G_x^0(z^*) \tag{7}$$

$$\overline{\overline{x}}^n = G_x^n(\uparrow \overline{\overline{x}}^{n-1}), \text{ for } n > 0 \tag{8}$$

$$\mathcal{L}_{recon_0}(G_x^0) = ||\overline{\overline{x}}^0 - x^0||_2 \tag{9}$$

$$\mathcal{L}_{recon_n}(G_x^n) = ||\overline{\overline{x}}^n - x^n||_2, \text{ for } n > 0 \tag{10}$$

The reconstruction loss is used to control σ^n, the standard deviation of the Gaussian noise used at each scale, z^n, which indicates the level of detail required at each scale. In particular, $\sigma^n = || \uparrow \overline{\overline{x}}^{n-1} - x^n||_2$. Without it (using uniform randomness), G_x^n can tint input images. The overall loss used at scale n, for some hyperparameter $\alpha > 0$, is then:

$$\min_{G_x^n} \max_{D_x^n} \mathcal{L}_{adv}(D_x^n, G_x^n) + \alpha \mathcal{L}_{recon_n}(G_x^n) \tag{11}$$

Anomaly score We cast the problem of anomaly detection as determining if the patches of a given test image, x_{test}, and their transformations, are real. That is, for a given test image x_{test}, and for each scale n, let $T_1(x_{test}^n), \ldots, T_M(x_{test}^n)$ be the result of applying M transformations on x_{test}^n, the down-sampled version of x_{test} at scale n. $D_x^n(T_i(x_{test}^n))$ can be viewed as a set of $H \times H$ vectors of size $M+1$, where each vector corresponds to a given patch of the input. By removing the 0'th element (the "fake" class) and applying softmax, the i'th element of this vector denotes the probability that this patch is of class i (the i'th transformation). Since D is trained on data from the normal class, the higher this value is, across all patches and scales, and for all transformations T_i, the higher our confidence in this sample being non-anomalous. Our anomaly score is, therefore, given as:

$$C_{\{x\}}(x_{test}) = \sum_{n=0}^{N} \sum_{i=1}^{M} \sum_{p \in H \times H} [D_x^{n*}(T_i(x_{test}^n))_p]_i \tag{12}$$

where D_x^{n*} outputs the last M maps of D_x^n, i.e., it outputs $H \times H$ vectors of size M, without the 0'th "fake" element. $D_x^{n*}(T_i(x_{test}^n))_p$ denotes the softmax probability vector of the p'the patch, and the index i is indexing this vector, to provide the pseudo-probability of the patch belonging to class (transformation) i. The lower $C_{\{x\}}(x_{test})$, the more anomalous x_{test} is.

3.2. From one shot to few shot anomaly detection

Moving to the few shot setting, we are now equipped with a subset $X_k = \{x_1, \ldots, x_k\}$ of X. To maximize our sample space, we would like to capture the inter-class variability offered by each of the x_i's. We, therefore, devise a model that captures the multi-scale patch distribution of each x_i. One possibility is to train a model for each x_i separately, and combine the scores of each model. However, this may be computationally expensive and time consuming. Instead, we use a single generative model that is conditioned on i.

Conditional generation To condition the generator on each training sample x_i, we concatenate a single channel, whose values are i everywhere, to the input z of the generator, obtaining an input, which we denote as $cat(z, i)$.

Let x be a tensor of dimensions $C \times H \times W$ (channels, height and width). We denote by $cat(x, i)$ the tensor of dimensions $(C+1) \times H \times W$, where the last channel equals i in all $H \times W$ positions. Generalizing our one-image generator, the generator trained on the set X_k is denoted by $G_{X_k}^n$ and is defined as follows:

$$\overline{x}_i^0 = G_{X_k}^0(cat(z^0, i)) \tag{13}$$

$$\overline{x}_i^n = G_{X_k}^n(cat(z^n + \uparrow \overline{x}_i^{n-1}, i)) + \uparrow \overline{x}_i^{n-1} \tag{14}$$

Training objectives Our losses extend those used in Sec. 3.1 to the few shot case. Let $D_{X_k}^n$ be the discriminator at scale n trained on X_k. We define $\mathcal{L}_{adv}^i(D_{X_k}^n, G_{X_k}^n)$ to be $\mathcal{L}_{adv}(D_{X_k}^n, G_{X_k}^n)$ (Eq. 6) applied with input x_i^n instead of x^n. $\mathcal{L}_{adv}^{multi}(D_{X_k}^n, G_{X_k}^n)$ is defined to be the sum over $\mathcal{L}_{adv}^i(D_{X_k}^n, G_{X_k}^n)$ for all i. The reconstruction loss for each sample x_i, at scale n, now becomes:

$$\overline{\overline{x}}_i^0 = G_{X_k}^0(cat(z^*, i)) \tag{15}$$

$$\overline{\overline{x}}_i^n = G_{X_k}^n(cat(\uparrow \overline{\overline{x}}_i^{n-1}, i)), \text{ for } n > 0 \tag{16}$$

$$\mathcal{L}_{recon_0}^i(G_{X_k}^0) = ||\overline{\overline{x}}_i^0 - x_i^0||_2 \tag{17}$$

$$\mathcal{L}_{recon_n}^i(G_{X_k}^n) = ||\overline{\overline{x}}_i^n - x_i^n||_2, \text{ for } n > 0$$

The full reconstruction loss $\mathcal{L}_{recon_n}^{multi}(G_{X_k}^n)$ is the sum over $\mathcal{L}_{recon_n}^i(G_{X_k}^n)$ for all i. Note that the generator, in the multi-shot case, learns to generate from multiple image distributions i. In other words, we have a set of k derived generators per scale n, each of which is given by $G_{X_k}^n(cat(\cdot, i))$. Similar to the one-shot case, the overall loss at scale n is:

$$\min_{G_{X_k}^n} \max_{D_{X_k}^n} \mathcal{L}_{adv}^{multi}(D_{X_k}^n, G_{X_k}^n) + \alpha \mathcal{L}_{recon_n}^{multi}(G_{X_k}^n) \tag{18}$$

The architecture of $D_{X_k}^n$ is not modified from the one-shot setting. The difference is that now, $D_{X_k}^n$ is trained on more samples X_k as well as generated samples for each sample i, and so captures a richer distribution. The anomaly score used in the $k > 1$ case, denoted $C_{X_k}(x_{test})$, is the same anomaly score of Eq. 12 with $D_{X_k}^n$ instead of D_x^n.

3.3. Defect Detection

We also consider the task of "defect detection" [14], which is a localized variant of anomaly detection. In this variant, normal samples are visually similar and anomalous samples contain subtle local changes. Since our method models the multi-scale patch distribution of each training image, it can be readily used to localize areas where anomalous features occur. For anomalous samples, only a small number of patches are anomalous, and the rest are similar to patches in normal samples. Therefore, instead of averaging over all patches, we average over the 5% of patches with the lowest anomaly score. Specifically, Eq. 12 is modified to:

$$C_{\{x\}}(x_{test}) = \sum_{n=0}^{N} \sum_{i=1}^{M} \sum_{p \in f_n^i(H \times H)} [D_x^{n*}(T_i(x_{test}^n))_p]_i$$

where $f_n^i(H \times H)$ denotes 5% of patch indices with lowest anomaly score for scale n and transformation i. The few shot score is defined with $D^n_{X_k}$ instead of D^n_x. The effect of using a different percentage is given in Sec. 4.3.

3.4. Architecture and training details

The generator $G^n_{X_k}$ and the discriminator $D^n_{X_k}$ each consist of five convolutional blocks. A convolutional block consists of: (i) a 3×3 convolutional layer and is padded such that it maintains the spatial resolution of the input, (ii) Batch normalization layer and (iii) LeakyReLU activation with a slope of 0.2. The application of five such blocks results in a fixed effective receptive field of 11×11 for generator $G^n_{X_k}$ and discriminator $D^n_{X_k}$ at each scale n. For the last convolutional block, we do not use batch normalization and a $tanh$ is used instead of a LeakyReLU. An adam optimizer learning rate of 0.0005 and parameters $\beta_1 = 0.5$ and $\beta_2 = 0.999$, are used. In Eq. 11 and Eq. 18, $\alpha = 100.0$ for all experiments. r is chosen to be 0.75 and N is chosen such that the maximal resolution at the finest scale N is 64×64.

4. Experiments

In all experiments, images are resized to a resolution of 64×64. For ten independent trials and for each class label separately, we train our model on k different images randomly selected from the normal class of images X. The same set of images are used for our method and for baselines. At test time, for each trial, using all test images, we measure the Area Under the Curve (AUC). We report the mean AUC and standard deviation values in the main text, and the detailed results per class in the supplementary.

4.1. Anomaly detection

We evaluate our method on four datasets: Paris [26], CIFAR10 [16], FashionMNIST [38] and MNIST [18]. The **Paris** dataset consists of 6,392 high resolution (1024×768) images obtained from Flicker and divided into 11 landmarks. **CIFAR10** consists of $60,000$ 32×32 color images in 10 classes, split to $50,000/10,000$ between train and test. **MNIST** and **FashionMNIST** both consist of $70,000$ 28×28 grayscale images of digits or fashion products, respectively, split to $60,000/10,000$ between train and test. There are 10 categories in each. Following [33, 3, 31], we consider as normal images, all images from a given class, and as anomalous images, the images of all other classes.

We compare our method to five recent baseline methods. The first method is GEOM [10] which applies different transformations to normal images and trains a classifier to classify the transformation applied. The second work is GOAD [3], which modifies the anomaly score used by GEOM. We also consider DROCC [11], that trains a classifier to distinguish the training samples from their perturbations generated adversarially. Lastly, we consider the method of DeepSVDD [31], which uses a similar objective to that of classic SVDD [34] together with features of a deep network, and PatchSVDD [39] which extends DeepSVDD to a patch-based method using self-supervised learning.

One shot anomaly detection Fig. 2(a) shows the result of our method in comparison to baselines. Our model outperforms all baselines for all datasets. It scores best for FashionMNIST and MNIST, where a single image captures significant inter-class variability. PatchSVDD, which incorporates a patch-based formulation, scores significantly lower, indicating that, using patches at different scales, together with a our generative and discriminative formulation, significantly improves results. DeepSVDD scores better then PatchSVDD, indicating that deep network features are important. DROCC, GEOAM and GOAD, which use a discriminative objective, are also inferior to our method, indicating that our patch based generative model is important.

Qualitative results for the Paris dataset on the one-shot setting are presented in Fig. 3, for typical samples from the "Defense", "Louvre" and "Pantheon" classes. For each class we train the model on random training image and show: (1) the training image, (2) two randomly generated samples, (3) true positive, true negative, false positive and false negative predictions. As can be seen, the generated images capture the appearance and texture of the class, but sometimes alter the structure. From the true positives, we observe that the model correctly classifies non trivial images that differ from the training image in color, orientation, zoom and even partial occlusions. The false positive (anomalous classified as normal) are often similar to the training sample. The false negatives are often images with significant occlusions or images where the landmark is very small and distant.

Few shot anomaly detection We evaluate our method both on $k = 5$ and $k = 10$ images. Fig. 2(b-c) shows the result of our method in comparison to baselines. Our model outperforms the baselines on all datasets, on both the five-shot and ten-shot settings. Increasing the number of samples boosts performance, in particular for the Paris dataset. where using five samples, instead of one, increased the AUC from 66.6% to 79.8%, indicating that multiple images are required to handle inter-class variability. On this dataset, our gap to baselines is also largest, indicating that our method better utilizes the additional variability provided by the additional samples. In Fig. 4 we consider the effect of increasing the number of samples used for training our method and baselines, on CIFAR10. Our method consistently improves with added number of samples and outperforms baselines on both the 50-shot and 80-shot settings. However, the gap from GEOM decreases as k increases, indicating that when many samples are available, the improvement gained from generating samples at multiple scales diminishes.

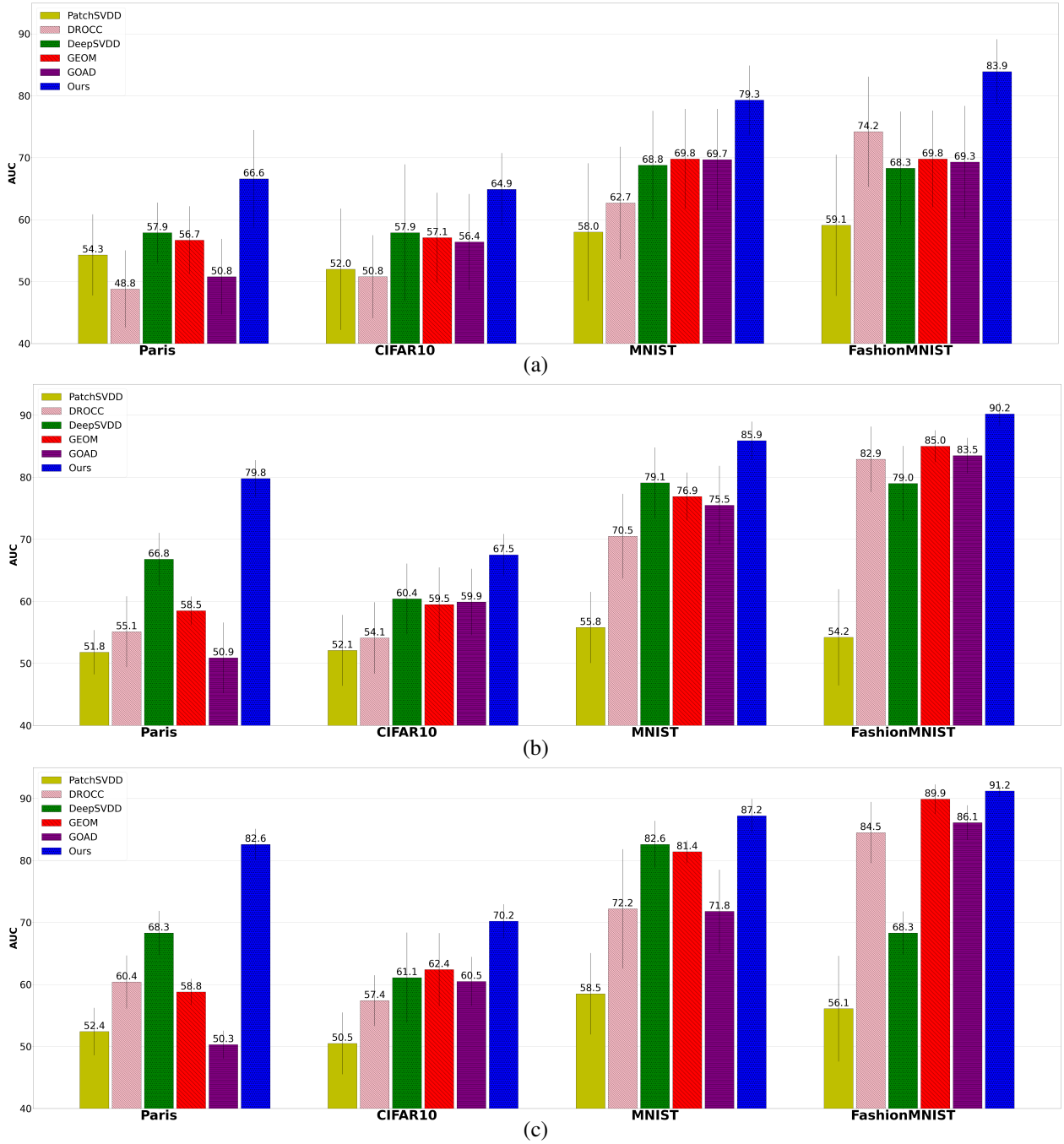


Figure 2. Average AUC (STD indicated with a vertical line) for one and few shot anomaly detection experiments on Paris, CIFAR10, FashionMNIST and MNIST datasets. (a): One Shot ($k = 1$), (b): Five Shot ($k = 5$) and (c): Ten Shot ($k = 10$).

4.2. Defect detection

For the task of defect detection, we evaluate our method on the MVTec dataset [4], which contains 5354 high-resolution color images of 10 object and 15 texture categories. These are split to 3629/1725 images between train and test. The number of training samples per category ranges from 60 to 320. A total of 70 defect types, such as little cracks, deformations, discolorizations and scratches occur in test images. The anomalies may differ in size, shape and structure. For each class, we consider the normal class to be all the defect-free images in this class, and the anomalous images to be all the defective images from the same class.

We compare our method to DifferNet [29] and PatchSVDD [39] that excel in defect detection. We also compare to anomaly detection methods described above. We

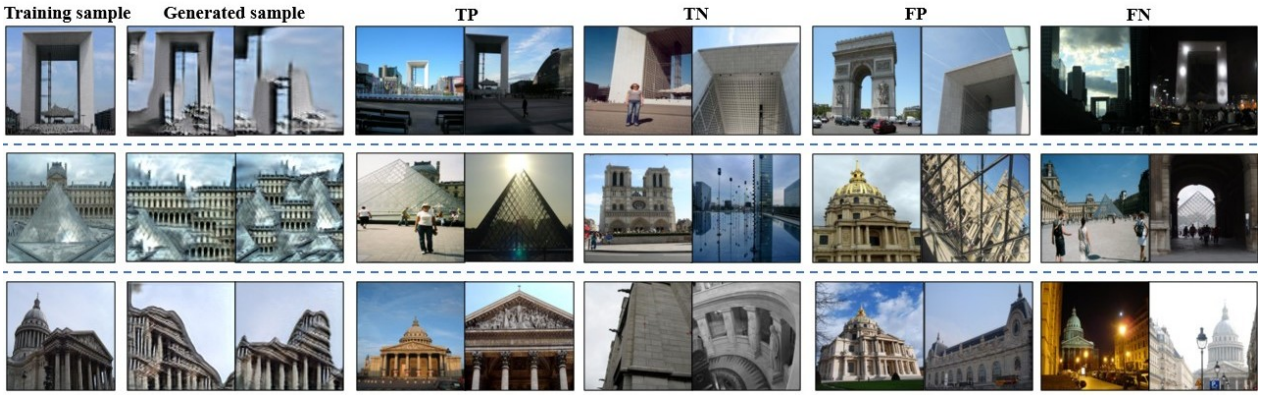


Figure 3. Illustration of classification decisions made by a single-shot model trained on the Paris dataset. The first column is the training sample, then are random samples generated by the trained generative model. The other columns present samples from the test set of the Paris dataset that are either true positive (**TP**), true negative (**TN**), false positive (**FP**) or false negative (**TN**).

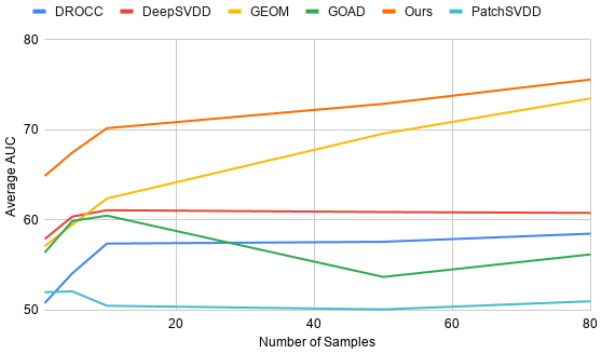


Figure 4. The effect of increasing the number of samples for our method and baselines on CIFAR10.

v.	G	T	H	1-shot	5-shot
Full	Yes	Yes	Yes	**64.9**	**67.5**
(a)	No	Yes	Yes	60.7	64.9
(b)	Yes	No	Yes	59.1	60.0
(c)	Yes	Augment	Yes	59.7	63.4
(d)	Yes	Yes	No (s=100)	57.6	57.8
(e)	Yes	Yes	No (s=20)	57.3	63.8
(f)	No	No	No	47.7	48.0
(g)	Generation followed by GEOM			58.8	63.7

Table 1. Average AUC for CIFAR10, with or without each of the main components of our method: (G) A generator model, (T) Employing transformations, (H) Hierarchy of patches. See more details in Sec. 4.3.

follow the same evaluation protocol for one-shot and few shot anomaly detection. We consider the standard set of transformations as described in Sec. 3.1 (Ours1 in Fig. 5). For a fair comparison with DifferNet we also consider only the four rotation transformations (group (4) in Sec. 3.1) as applied in DifferNet (Ours2 in Fig. 5). As can be seen in Fig. 5, our method outperforms all baselines.

Fig. 6 illustrates the localization of the defects for one-shot defect detection, for both our method and for DifferNet. Three random test samples from MVTec are shown. We visualize the defects at the final scale. Following the notation of Sec. 3.3, our visualization map is defined as $\sum_{i=1}^{M} T_i^{-1} \left[D_x^{N*}(T_i(x_{test}^N))_{(:)}\right]_i$, where $\left[D_x^{N*}(T_i(x_{test}^N))_{(:)}\right]_i$ is an $H \times H$ map indicating how real $T_i(x_{test}^N)$'s patches are. We consider only the patch indices in $f_i^N(H \times H)$, and set the other patches to 0. For DifferNet, we use the visualization procedure provided in their work. Our method accurately captures the defect regions whereas DifferNet only captures smaller regions of the defect area.

4.3. Ablation analysis

Our method relies on three main components: (1) a generative model, (2) its hierarchical multi-scale nature, and (3) a transformation-discriminating component. We assess the contribution of these components separately, running ablation experiments on CIFAR10 for both one-shot and five-shot settings. A first variant (variant (a), or v.(a) for short) does not have a **generative component (G)**. Instead, we only use D_n^x, and remove the fake class 0. D_n^x is trained to classify between real images at this scale and their transformations (no fake images are used). The anomaly score remains the same. The second variant (v.(b)) does not employ **transformations discriminatively (T)**. D_n^x is trained to distinguish between real and fake images at scale n, and not between transformations of images. This is equivalent to setting $M = 1$ and using T_1 as identity. As another alternative (v.(c)), we apply $T_1, \ldots, T_M$ as augmentations (uniformly at random) before being fed to D_n^x. The next variants consider a single scale of the **hierarchy (H)**, by setting $N = 0$. We use the same 11×11 receptive field, and downscale the image to either 100×100, where small patches are considered, or to 20×20, where large patches are considered. This is indicated by $s = 100$ (v.(d)) and $s = 20$ (v.(e)). We also consider a simple baseline where no

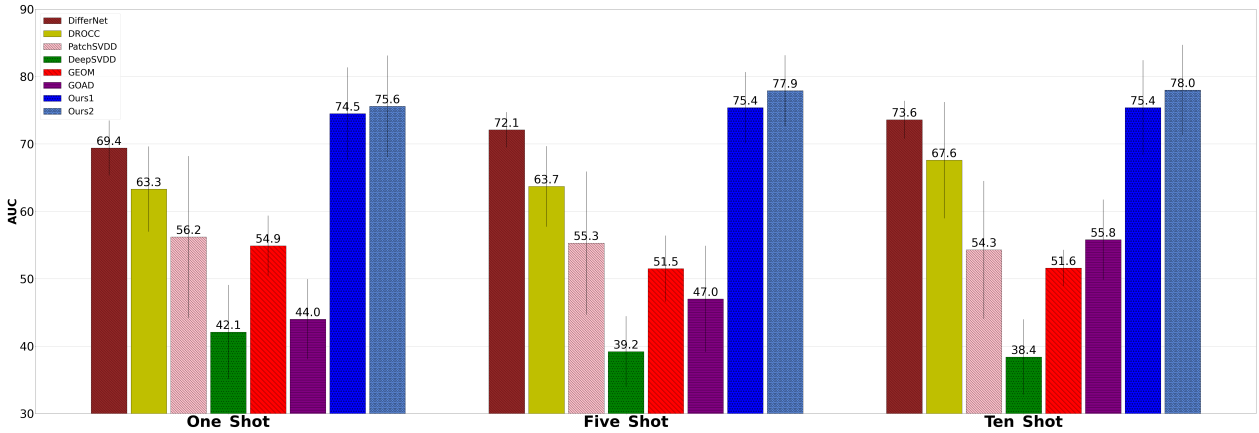
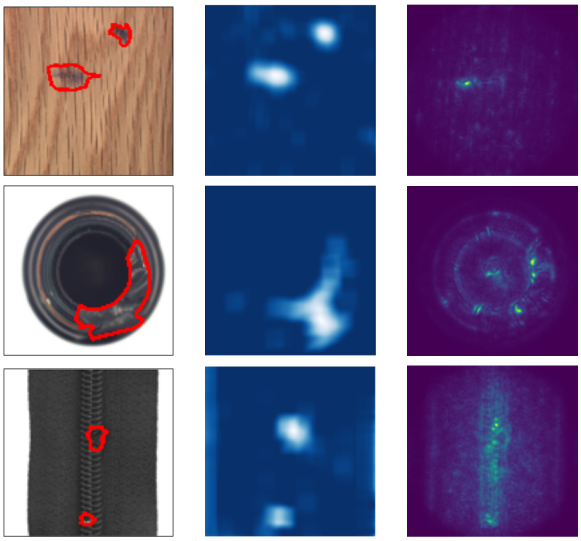


Figure 5. Average AUC (STD indicated with a vertical line) for defect detection on MVTec, for the **One Shot**, **Five Shot** and **Ten Shot** settings. For **Ours1**, the transformations of anomaly detection (Sec. 3.1) are used, while for **Ours2**, only rotations are used (as in DifferNet).



(a) (b) (c)

Figure 6. Localization of defects in MVTec test images for one-shot defect detection. (a) The original images, in which the anomaly region is delineated in red. (b-c) The localization provided by (b) our method and (c) DifferNet.

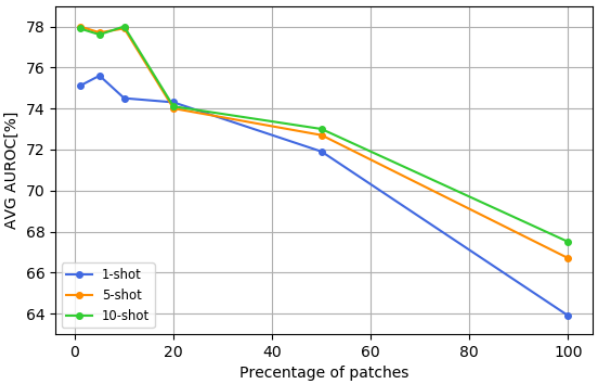


Figure 7. Effect of percentage of patches used for defect detection.

component is used. The anomaly score is the MSE between the test image and the training image in the one-shot setting, and the average MSE in the five-shot setting (v.(f)). Finally, variant v.(g), trains a GEOM [10] model on $6,000$ samples generated using our generative model.

The results are reported in Tab. 1. All three components (G, T and H) are required to achieve best performance, with H (hierarchy of patches) playing a particularly important role. Applying GEOM to images generated by out networks is also not as effective as our method. However, it is more effective than running GEOM in the few shot setting (Fig. 2).

We further analyze the effect of the percentage of patches ($f_n^i(H \times H)$) taken for defect detection (See Sec. 3.3). Fig. 7 gives the average AUC for MVTec as a function of percentage of patches. Using 5% of patches is best for the one-shot setting while 10% is best for five and ten shot settings. However, the results are stable, when this ratio remains low.

5. Conclusions

We present a multi-scale hierarchical generative model, which incorporates, within the discriminators, the self-supervised task of classifying transformations. While multiclass descriptors, in the supervised case, are common in conditional GANs, e.g., [32], we are not aware of other methods that combine labels from a SSL task. Also, while discriminators play an important role in adversarial learning, most works do not employ them outside of training a generator or for creating a secondary feature matching loss [27]. Also unique, as far as we can ascertain, is the training of single-image like GANs on multiple images. This is done by adding a conditioning layer that contains the image index.

Our method presents a very sizable gap in performance in comparison to the state of the art methods for the few-shot case. Admittedly, training becomes more involved with the increase in the number of training images, and the method does not scale well to hundreds or thousands of training im-

ages, without further modifications. In the case of a small training set, for which our method was designed, it demonstrates one-class classification capabilities that are surprising given the emphasis in the existing literature on modeling the form of the variability between the training samples.

References

[1] Charu C Aggarwal. Outlier analysis. In *Data mining*, pages 237–263. Springer, 2015.

[2] Jinwon An and Sungzoon Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2(1):1–18, 2015.

[3] Liron Bergman and Yedid Hoshen. Classification-based anomaly detection for general data. *arXiv preprint arXiv:2005.02359*, 2020.

[4] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9592–9600, 2019.

[5] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

[6] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*, 2019.

[7] Li Fei-Fei. Knowledge transfer in learning to recognize visual objects classes. In *Proceedings of the International Conference on Development and Learning (ICDL)*, page 11, 2006.

[8] Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006.

[9] Ahmed Frikha, Denis Krompaß, Hans-Georg Köpken, and Volker Tresp. Few-shot one-class classification via meta-learning. *arXiv preprint arXiv:2007.04146*, 2020.

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

[11] Sachin Goyal, Aditi Raghunathan, Moksh Jain, Harsha Vardhan Simhadri, and Prateek Jain. Drocc: Deep robust one-class classification. In *International Conference on Machine Learning*, pages 3711–3721. PMLR, 2020.

[12] Shir Gur, Sagie Benaim, and Lior Wolf. Hierarchical patch vae-gan: Generating diverse videos from a single sample. In *NeurIPS*, 2020.

[13] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *Advances in Neural Information Processing Systems*, 33, 2020.

[14] Christian Koch, Kristina Georgieva, Varun Kasireddy, Burcu Akinci, and Paul Fieguth. A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure. *Advanced Engineering Informatics*, 29(2):196–210, 2015.

[15] Jedrzej Kozerawski and Matthew Turk. Clear: Cumulative learning for one-shot one-class image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3446–3455, 2018.

[16] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *University of Toronto*, 2009.

[17] Anna Kruspe. One-way prototypical networks. *arXiv preprint arXiv:1906.00820*, 2019.

[18] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. *Online*, 2010.

[19] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *European conference on computer vision*, pages 702–716. Springer, 2016.

[20] Mary M Moya, Mark W Koch, and Larry D Hostetler. One-class classifier networks for target recognition applications. *NASA STI/Recon Technical Report N*, 93:24043, 1993.

[21] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? *arXiv preprint arXiv:1810.09136*, 2018.

[22] Guansong Pang, Longbing Cao, Ling Chen, and Huan 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*, pages 2041–2050, 2018.

[23] Guansong Pang, Chunhua Shen, Huidong Jin, and Anton van den Hengel. Deep weakly-supervised anomaly detection. *arXiv preprint arXiv:1910.13601*, 2019.

[24] Guansong Pang, Chunhua Shen, and Anton van den Hengel. Deep anomaly detection with deviation networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 353–362, 2019.

[25] Guansong Pang, Anton van den Hengel, Chunhua Shen, and Longbing Cao. Deep reinforcement learning for unknown anomaly detection, 2020.

[26] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *2008 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2008.

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

[28] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *Computer Vision (ICCV), IEEE International Conference on*, 2019.

[29] Marco Rudolph, Bastian Wandt, and Bodo Rosenhahn. Same same but differnet: Semi-supervised defect detection with normalizing flows. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1907–1916, 2021.

[30] Lukas Ruff, Jacob R Kauffmann, Robert A Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G Dietterich, and Klaus-Robert Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 2021.

[31] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *International conference on machine learning*, pages 4393–4402, 2018.

[32] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*, 2016.

[33] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International conference on information processing in medical imaging*, pages 146–157. Springer, 2017.

[34] Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. *Advances in neural information processing systems*, 12:582–588, 1999.

[35] Assaf Shocher, Shai Bagon, Phillip Isola, and Michal Irani. Ingan: Capturing and retargeting the "dna" of a natural image. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019.

[36] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? *arXiv preprint arXiv:2003.11539*, 2020.

[37] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)*, 53(3):1–34, 2020.

[38] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[39] Jihun Yi and Sungroh Yoon. Patch svdd: Patch-level svdd for anomaly detection and segmentation. In *Proceedings of the Asian Conference on Computer Vision*, 2020.

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

[41] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. *arXiv preprint arXiv:2006.10738*, 2020.

[42] Yang Zhou, Zhen Zhu, Xiang Bai, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Non-stationary texture synthesis by adversarial expansion. *ACM Trans. Graph.*, 37(4):49:1–49:13, July 2018.

[43] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.