# Zero-shot versus Many-shot: Unsupervised Texture Anomaly Detection

Toshimichi Aota
Sanoh Industrial Co., Ltd.
t.aota@sanoh.com

Lloyd Teh Tzer Tong
Sanoh Industrial Co., Ltd.
t.lloydteh@sanoh.com

Takayuki Okatani
Tohoku University / RIKEN AIP
okatani@vision.is.tohoku.ac.jp

## Abstract

*Research on unsupervised anomaly detection (AD) has recently progressed, significantly increasing detection accuracy. This paper focuses on texture images and considers how few normal samples are needed for accurate AD. We first highlight the critical nature of the problem that previous studies have overlooked: accurate detection gets harder for anisotropic textures when image orientations are not aligned between inputs and normal samples. We then propose a zero-shot method, which detects anomalies without using a normal sample. The method is free from the issue of unaligned orientation between input and normal images. It assumes the input texture to be homogeneous, detecting image regions that break the homogeneity as anomalies. We present a quantitative criterion to judge whether this assumption holds for an input texture. Experimental results show the broad applicability of the proposed zero-shot method and its good performance comparable to or even higher than the state-of-the-art methods using hundreds of normal samples. The code and data are available from* `https://drive.google.com/drive/folders/10OyPzvI3H6llCZBxKxFlKWt1Pw1tkMK1`.

## 1. Introduction

In this paper, we consider the problem of detecting anomalies of a texture from its image. As it is an important problem that frequently occurs in industry, there have been many studies in computer vision [19, 15]. Recent studies formulate it as a problem of unsupervised learning based on real-world requirements. Specifically, assuming the availability of a certain amount of normal images, we wish to detect anomalies that were unseen before.

After some trials were conducted [19, 14], it was recently found to be effective to use the intermediate features of CNNs (or ViT [10]) pre-trained on ImageNet. Using them with simple distance-based classifier [6, 16, 7, 17] outperforms other approaches by a large margin, e.g., reconstruction-based methods using autoencoders etc. [1, 3, 8]. This 'rediscovery' of the off-the-shelf visual features has pushed the detection performance on MVTec AD [1], a standard benchmark dataset for anomaly detection, to nearly perfect detection accuracy. Therefore, anomaly detection (AD) research will be shifting its target to more challenging problems. One of the possible research directions is few-shot AD, i.e., detecting anomalies in the condition that only a few normal images are available.

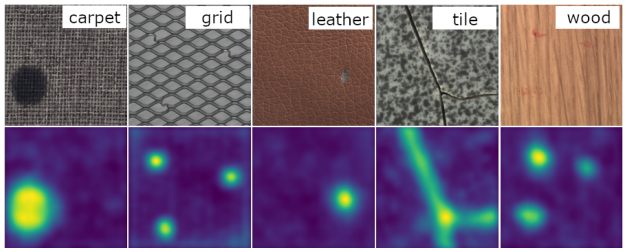


Figure 1. Examples of texture images from the MVTec AD datasets [1] and their anomaly scores computed by the proposed zero-shot method.

In this paper, we consider how few normal samples are needed for accurate texture AD. We first highlight the critical nature of the problem that previous studies have overlooked. Namely, AD gets more challenging when the input texture is anisotropic (i.e., having an oriented pattern) and when the orientations of input and normal images are not aligned. Its impact is more significant in fewer-shot settings. There are multiple solutions, e.g., increasing the number of normal samples or aligning the image orientation. The latter strategy is employed in a recent study of few-shot AD [12]; however, it raises another difficulty of aligning the orientation of the images.

We next propose a zero-shot method, which detects texture anomaly without using a normal sample. As it does not compare the input with any normal sample, it is free from the above issue with image orientation. The method uses the same pre-trained CNN features as the existing anomaly detection methods mentioned above.

Why is zero-shot AD feasible? Considering that AD is to find images/regions that are different from normal samples, it seems impossible to do it without knowing what is normal. The answer is that our method solves it by con-

verting it into another problem. Namely, it detects an image region that breaks the homogeneity of the input image. This principle may be analogous to how humans detect anomalies from only a single image without much effort; see the images of Fig. 1. An underlying assumption is that the input textures, if anomaly-free, have a certain level of homogeneity in their local appearance. We propose a quantitative criterion to check if this assumption holds for an input texture. It enables us to judge the effectiveness of the proposed zero-shot AD method in advance.

We report the results of experiments conducted to verify our approach. They show that the proposed zero-shot method achieves an average image-level AUROC of $99.6\%$ over the five textures in MVTec AD. This performance is comparable to PatchCore [17], the SOTA method for the dataset, which achieves $99.0\%$ (AUROC) over the same five textures when using hundreds of normal images; its performance decreases to 93.0% in a five-shot setting. For more detailed analysis, we create a dataset named *DTD-Synthetic* by borrowing diverse texture images from DTD (Describable Texture Dataset) [5] and synthesizing diverse anomalous patterns. The results verify the above argument on texture orientation. They show that our zero-shot method achieves a detection accuracy comparable to PatchCore in a 100-shot setting, i.e., image-level AUROC 98.9% vs. 98.0%. We also show the results on the Aitex fabrics datasets [21] and DAGM2007 [22], which also verify the effectiveness of our approach.

## 2. Related Work

The problem of detecting anomalies from images has a long history of research [4, 19, 14]. Due to lack of space, we consider only recent studies of unsupervised AD here. The recent studies can be classified in terms of two factors. One is how to extract features from input images. The other is how to represent normal samples in the space of the extracted features to identify anomalies.

There are two approaches to the first factor, feature extraction. One is to learn to extract features fit for AD. It may further be divided into reconstruction-based methods [20, 3, 8] and methods based on self-supervised and/or metric-learning [2, 13]. These methods usually need a certain amount of normal samples for training. The other approach employs the features of pre-trained networks [6, 16, 7, 17]. It does not need training on normal samples, so it is more fit for few-shot AD.

There are also several approaches to the second factor, i.e., how to represent normal samples. The simplest is the gallery-based approach, which constructs a gallery of normal samples in the feature space, and computes the distance from each input to its nearest neighbor(s) in the gallery to detect anomalies. Another approach uses a parametric distribution like a Gaussian distribution to model the distribution of normal samples in the feature space [7]. Some studies use more flexible model like normalizing flow to represent the normal samples' distribution [18, 23, 11]. These methods use the distribution to detect anomalies, e.g., by computing and thresholding the probability that the input is normal. Others employ a teacher-student framework [2, 9], in which a teacher network is distilled to student network(s) with its response to normal samples. They judge whether an input is normal by checking the consistency in behavior between the teacher and students.

While all these methods attain good accuracy in the standard benchmark, such as MVTec AD [1], the gallery-based methods are the fittest for few-shot settings. The others need a certain amount of normal samples to represent normal samples' distribution accurately. The representative methods of the gallery-based methods are SPADE [6] and PatchCore [17]. They employ pre-trained CNN features (or ViT [10]) for the feature extraction. Thus, they are training-free and ready to be applied to few-shot settings. We will consider PatchCore, which achieves state-of-the-art performance on the MVTec AD dataset, for comparison with our zero-shot method.

As far as the authors know, there are only a few studies considering unsupervised AD in few-shot settings [12, 17]. RegAD [12] is designed for the few-shot AD; it geometrically transforms input images into a canonical pose to cope with the difficulties specific to few-shot AD. We may regard it as a solution to cope with the above issue of unaligned image orientations. The present study proposes a zero-shot method as another solution. It is noteworthy that RegAD does not necessarily perform better than PatchCore in few-shot settings, as will be seen later.

## 3. Anomaly Detection with Anisotropic Texture

As mentioned above, the gallery-based methods are ready in principle to be applied to few-shot AD. In fact, they achieve fairly good performance, especially for textures. However, there are some complexities in their behavior.

Table 1 shows the results of few-shot AD by PatchCore for the five MVTec AD textures. We can see that the few-shot detection accuracy varies greatly among these textures. As compared with the case of using all available samples, few-shot accuracy deteriorates significantly for *grid* and modestly for *wood*, respectively. On the other hand, it decreases only slightly for *carpet* and *tile*; there is even no change (i.e., 100% for 1-shot) for *leather*. What leads to such differences?

There are a few factors leading to these differences. The most vital one is the isotropy of textures, i.e., whether or not a texture has an orientation. For the MVTec AD textures, *carpet*, *grid*, and *wood* are anisotropic, whereas *leather* and

Table 1. Performance (image-level AUROC) of PatchCore [17] on five textures (see Fig. 1) of MVTec AD [1] in few-shot settings.

| Num. of Shots | 1 | 5 | 10 | 100 | All |
|---|---|---|---|---|---|
| carpet | 98.6 | 98.7 | 98.6 | 98.8 | 98.6 |
| grid | 52.3 | 69.5 | 82.7 | 96.6 | 97.3 |
| leather | 100 | 100 | 100 | 100 | 100 |
| tile | 98.4 | 98.5 | 98.6 | 98.6 | 98.9 |
| wood | 98.3 | 98.6 | 98.6 | 98.8 | 99.4 |

*tile* are isotropic, or only slightly anisotropic. See Fig. 1 for examples of the five textures.

This difference in isotropy causes the above differences in few-shot performance. For example, suppose an anisotropic texture like *grid*. If its input image has a different orientation from the normal samples, its local regions will have different features from any stored ones in the gallery. Then, the gallery-based methods will classify every local region wrongly as anomalous even when it is not.

Even in that case, when the normal samples have diverse orientations, one of them may happen to have a similar orientation to the input. If so, the gallery-based methods can classify the input correctly. Assuming that the normal samples' orientations are random, the chance of finding a normal sample having a similar orientation will increase with the growing number of normal samples. This analysis agrees well with the results for *grid*; the accuracy increases with the number of shots (i.e., normal samples). This also holds for *wood*, while it is less significant.

However, this explanation does not hold for *carpet*; while it is an anisotropic texture, few-shot cases achieve similar accuracy to the case of all samples. Why? It is because that *carpet*'s samples in the dataset are aligned to have the same orientation. Since test samples have similar features to normal samples, the accuracy of few-shot (even one-shot) detection is good.

Note that the above is a fundamental issue with unsupervised AD; it will similarly affect other types of methods, while we use the gallery-based methods for the explanation. Furthermore, although it is more noticeable in few-shot settings, it can affect the performance in many-shot settings depending on conditions (e.g., the strength of texture anisotropy, average difference in image orientation, and the number of normal samples).

# 4. Zero-shot Anomaly Detection

We next present a zero-shot method for texture AD. As it does not use a normal sample, it is free from the above issue of texture orientation.

## 4.1. Revisiting State-of-the-art AD Methods

Before explaining our method, we revisit the gallery-based methods for unsupervised AD, including SPADE [6] and PatchCore [17]. Assuming a set of normal images, $\mathcal{D} = \{x_n\}_{n=1,\dots,N}$, these methods use distances to each sample in $\mathcal{D}$ measured in the feature space to judge whether an input $x$ includes anomaly or not. For the image feature, most of them employ local image features extracted from $x$ using a pre-trained network (e.g., ResNet-50 or ViT [10]).

More details are as follows. Let $f_{ij}^l \in \mathbb{R}^{C^l}$ be the vector comprising the $l$-th layer activation having the shape $W^l \times H^l \times C^l$ at its spatial coordinate $(i, j)$ ($i = 1, \dots, W^l$, $j = 1 \dots, H^l$). Existing methods use the concatenation of different layer features to form a feature vector $f_{ij}$ at $(i, j)$. PatchCore further employ *locally aware patch features* instead of the original layer features. Choosing relatively lower layers for extracting feature $f_{ij}$, it applies local average pooling with $s \times s$ to them to obtain pooled features, which are used as $f_{ij}$'s.

Then, they create the gallery $G$ of normal image features, i.e., the set of $f_{ij}^l$'s of normal images. SPADE creates $G$ on demand for an input image $x$ by finding several nearest images to $x$ in $\mathcal{D}$. PatchCore finds the core-set of normal image features in an offline manner and creates $G$. To detect anomaly for a given image $x$, SPADE and PatchCore extract $\{f_{ij}\}$ for $x$ and search for the nearest neighbor $f$ to $f_{ij}$ at each $(i, j)$ in $G$. If their distance $\|f - f_{ij}\|_2$ is higher than a pre-defined threshold, $x$ is judged to contain an anomaly at $(i, j)$.

## 4.2. Proposed Zero-shot Method

**Outline of the Method** Now we consider detecting an anomaly in a zero-shot manner from an image of textures. As explained in Sec. 1, we reformulate the problem as follows: we regard the image as anomalous *if there is any region that appears differently from others*, and normal otherwise. This is also restated as: *whether the texture in the image is homogeneous*. Strictly, this formulation is only effective for image-level anomaly detection, i.e., detecting if the input image is anomalous as it contains an anomalous region. However, this will be applicable to pixel-level detection under a mild assumption, as will be discussed later.

To perform the above judgment for an input image $x$, we extract features $f_{ij}$ from it using a pre-trained CNN as in the above methods; the details will be given later. Then, we consider evaluating how similar the features $f_{ij}$'s at different image points $(i, j)$'s are. Among several candidate methods for doing this, we consider judging if the feature $f_{ij}$ at each image point $(i, j)$ has *many* similar features at other image points. To do this, we create a feature gallery $G$ as

$$G = \{f_{ij} \,|\, i = 1, \dots, W^l, j = 1, \dots, H^l\}. \quad (1)$$

Note that $G$ could contain anomalous vectors here unlike the above. Now, to conduct the above judgment, we compute the average distance $d_{ij}$ from $f_{ij}$ to its $K$ nearest

neighbors in $G$ as

$$d_{ij} = \frac{1}{K} \sum_{f \in N_p(f_{ij})} \mathrm{dist}(f_{ij}, f), \tag{2}$$

where $N_p(f_{ij})$ is the $K$ nearest neighbors. Then, if there exists $(i, j)$ such that $d_{ij}$ is large, the image $x$ is anomalous; otherwise, it is anomaly-free (i.e., homogeneous).

**Pixel-level Anomaly Detection** The above method is designed for image-level anomaly detection, which suffices for most real-world applications. However, under a mild assumption, the same method can also be used for pixel-level anomaly detection. The assumption is that *normal regions are the majority and the anomalous ones are the minority*. It holds in most practical cases, as shown in our experiments. Then, we can use $d_{ij}$ defined above directly as an 'anomaly score' of the local region at $(i, j)$, immediately enabling pixel-level anomaly detection. Note that even when the assumption does not hold, our method can still perform image-level detection correctly; this is the case even when anomalous regions dominate an image since our method judges the texture homogeneity of the input image.

**Details of Feature Extraction** The details of extracting $f_{ij}$ from $x$ is as follows. Selecting a single layer $l$, we extract its activation of size $W^l \times H^l \times C^l$. Following PatchCore, we apply the local average pooling to it. We then obtain the feature vectors $\{f_{ij}\}$ ($i = 1, \ldots, W^l$, $j = 1, \ldots, H^l$). It should be noted that unlike the above methods, we do not apply any dimensionality reduction to the feature vectors. It is not necessary in our case since the size of gallery $G$ is small and thus the computational cost with searching over $G$ is small as well.

As with other standard CNNs, the employed CNN perform zero-padding at each convolution layer. This makes the feature vectors $f_{ij}$'s at the image boundary dissimilar to the rest of the feature vectors even when the textures are similar. This could results in judging the image periphery to be anomalous even if it is not. To cope with this, we follow the existing methods and apply center-crop to the anomaly score map. We can specify its size based on the size of the receptive field for the convolution filter at layer $l$. Note that the existing methods employ center cropping as well. Due to the nature of zero-shot detection, our method tends to need a crop of slightly smaller center region. Note also that this procedure may not be necessary when we employ ViT.

We then resize $d_{ij}$ to match the original resolution of the input image $x$ using bilinear interpolation. We optionally apply Gaussian filtering to the resized map to remove noises. Let $\bar{d}_{ij}$ ($i = 1 \ldots, W, j = 1, \ldots, H$) be the resulting map. We judge whether a pixel $(i, j)$ is anomalous or not by simply comparing $\bar{d}_{ij}$ with a threshold. Changing this threshold, we obtain a ROC curve, from which we compute image-level and pixel-level AUROC.

### 4.3. Predicting Applicability of the Method

As explained above, our method judges the homogeneity of an input texture. An underlying assumption is that the input texture is homogeneous if it is anomaly-free. In other words, our method is only applicable to textures that meet the assumption. Therefore, it will be ideal if we can judge the method's applicability in advance for individual textures.

We propose to use the maximum of the anomaly scores computed for each image, i.e.,

$$\alpha(I) \equiv \max_{1 \le i \le W, 1 \le j \le H} \bar{d}_{ij}, \tag{3}$$

where $I$ is an input image. Concretely, assuming we are given an anomaly-free image $I$ of the texture of interest, we compute $\alpha(I)$ and compare it with a pre-defined threshold. We judge that the method can detect anomalies reliably if $\alpha(I)$ is lower than the threshold. We will show the effectiveness of this criterion in Sec. 5.5.

The quantity $\alpha(I)$ captures a kind of homogeneity of the brightness structure of $I$. Roughly speaking, if $\alpha(I)$ is small, then $I$ will look similar locally at any image location, and vice versa. Then, $\alpha(I)$ will be large for images of objects, such as those contained in MVTec AD, and it will be small for many texture images. More precisely, $\alpha(I)$ will be small for textures with homogeneous random structure or with a small-scale repetitive structure. For the latter, even if the texture has a precise repetitive structure, $\alpha(I)$ will be large when $I$ is a close-up of the texture, in other words, when the structure repeats for only a few counts in $I$. These agree well with how and why the proposed method works, underwriting the effectiveness of $\alpha(I)$.

However, we must use the above criterion with some consideration. First, since we employ a ReLU network for feature extraction, when we multiply the image brightness of $I$ by a scalar $a$, all the feature vectors and thus their distances will automatically be multiplied by $a$, assuming the network is bias-free. This highlights the importance of the normalization of input images. We employ the same normalization method as the previous studies, which use the mean and variance of brightness over the training split of the dataset. In addition, while our method detects anomalies by thresholding the anomaly score $\bar{d}_{ij}$, we judge the method's applicability by thresholding the maximum $\alpha(I)$. This means that the criterion implicitly assumes that the scores of the anomalies should lie in a specific range. We leave the validation of these to the experiments in Sec. 5.5.

## 5. Experimental Results

### 5.1. Experimental Settings

**Network and Hyperparameters** Following previous studies [6, 7, 17], we choose WideResnet-50-2 [24] pre-

trained on ImageNet for feature extraction. Following PatchCore, we choose a relatively lower layer for extracting features; specifically, we choose the output from the second block of the four in total. Following the previous studies, we first resize the input images to a fixed resolution. Considering the above effects of zero-padding at the image boundaries, we resize the images into slightly higher resolution (i.e., $320 \times 320$) than the previous studies (i.e., $256 \times 256$). Then, the resolution for the chosen layer is $40 \times 40$. As a result, the feature gallery $G$ contains 1,600 vectors of size 512. They are locally average-pooled with patch size $3 \times 3$ ($s = 3$).

We then calculate the anomaly score map for the layer according to Eq. (2). For $K$, the number of the nearest neighbors in the gallery, we set $K = 400$. As long as $K$ is large, the results are not sensitive to its choice. See the supplementary material for detailed analysis. We resize the computed anomaly score map to the input size, i.e., $320 \times 320$. After applying a Gaussian filter with $\sigma = 4$, we center-crop the anomaly score map with size $256 \times 256$, which is the final output of our method.

It should be noted that the previous studies choose $256 \times 256$ for the input size and $224 \times 224$ for the center crop, whereas ours are $320 \times 320$ and $256 \times 256$, respectively. We choose the configuration to cope with the boundary effect that are more severe with the zero-shot setting, as mentioned above. As with previous studies, we apply the same resize and center-crop to the ground truth detection masks, which makes our evaluation slightly different from the previous studies. To make fair comparisons, we evaluate the performance of SPADE and PatchCore in the same settings in our experiments. We follow the original papers for their hyperparameters. We choose 25% for the memory bank subsampling level in PatchCore throughout the experiments.

**Metrics of Detection Accuracy** We follow the previous studies to evaluate image-level and pixel-level detection accuracy. They are measured by AUROC of the detection of anomalous images and pixels, respectively. We consider the image-level AUROC for the primary metric to evaluate methods since it will be more important from a practical point of view. Considering the size of input images is relatively small, practitioners will find it more essential to be able to accurately detect images containing anomalies than to segment anomalous pixels inside images.

## 5.2. Datasets

**Existing Datasets** We use the five texture classes in the MVTec AD dataset [1] (Sec. 5.3). There are about 240 normal sample images for each class, whose resolution ranges from $700 \times 700$ to $1,024 \times 1,024$ pixels. We resize each image into $320 \times 320$ pixels and crop its center with $256 \times 256$ pixels for anomaly detection and its evaluation, as explained above. We also test our approach on the Aitex fabrics dataset [21] and the DAGM2007 dataset [22] (Sec. 5.6).

**DTD-Synthetic** To verify our arguments in Sec. 3 and test the proposed zero-shot method on more diverse data, we create a new dataset. We choose DTD (Describable Texture Dataset) [5], which were created for the research of texture classification, because of its diversity of textures. Borrowing the images of DTD, we synthesize texture images with anomalies. We call the dataset *DTD-Synthetic* in what follows.

DTD consists of 47 texture classes, each of which includes 120 different texture images, and thus 5,640 different texture images in total. We choose twelve images from them that are fit for the purpose here. Their resolution ranges from $300 \times 300$ to $640 \times 640$ pixels. To confirm our argument on texture orientation, we generate multiple images by cropping a square region of 60% width and height from the original image with a random *orientation* and position; see Fig. 2. This also simulates the image acquisition at factories etc. well. Thus, the resulting images for each texture have the size ranging from $180 \times 180$ to $384 \times 384$ depending on their original image.

We then synthesize anomalous patterns in these images, as shown in Fig. 2. We consider five types of anomalous patterns, i.e., line, color, size, bend, and shape, to simulate industrial inspection scenarios following MVTec AD. For each image, choosing one of the five class randomly, we draw a single instance of synthesized anomalies into a random position of a normal image generated as above [1].

We classify the chosen twelve textures into the following three categories. Each category contains four different textures.

**Category-1: Anisotropic textures having a repetitive structure with perfect regularity.** They are similar to *grid* in MVTec AD and are often found on the surfaces of man-made objects made of hard materials.

**Category-2: Anisotropic textures having a repetitive structure with some irregularity.** They are similar to *carpet* and wood in MVTec AD and are usually found on textiles and surfaces of natural objects.

**Category-3: Isotropic textures without obvious repetitive patterns.** They are similar to *tile* and *leather* of MVTec AD; they do not have a clear repetitive structure. Their local structure is identical at any position and orientation.

There are 12 textures in total. For each of them, we generate 100 train (i.e., anomaly-free) images and over 100 test images. The latter contains about 80 anomaly images and 20 or more normal sample images. Details are shown in the

[1] The dataset is downloadable from: `https://drive.google.com/drive/folders/10OyPzvI3H6llCZBxKxFlKWt1Pw1tkMK1`.

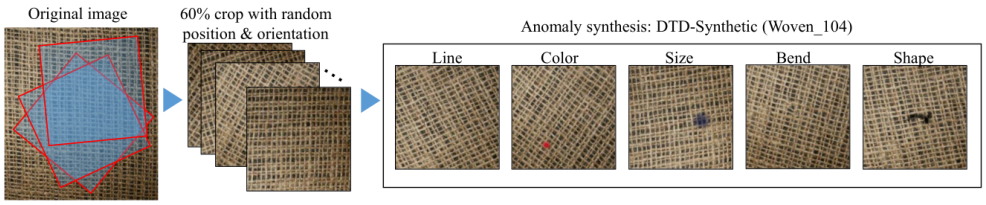


Figure 2. Method for creating normal and anomalous texture images of DTD-Synthetic. For a selected image from DTD, a square region with its 60% size is cropped with random positions and orientations. Then, five classes of synthesized anomalous patterns (i.e., line, color, size, bend, and shape) are drawn into a random position of the cropped images.

supplementary materials. Note that the proposed zero-shot method does not need the train images.

### 5.3. Results on MVTec AD

We first show the performance of our zero-shot method on the five textures of MVTec AD. Table 2 shows its image-level AUROC along with those of SPADE, PatchCore, and RegAD. It can be seen that our method achieves good performance for any texture, which is comparable to the SOTA methods, without a single normal sample image. It is noteworthy that RegAD, a method dedicated for few-shot settings, does not necessarily perform better than PatchCore in similar settings. (It should be noted that the results for ours, SPADE, and Patchcore are obtained in the same experimental settings and those for RegAD in slightly different settings.)

As mentioned in Sec. 3, PatchCore (and SPADE) shows lower performance for *grid* and *wood* in few-shot settings. Using more shots leads to better performance. It is interesting that RegAD shows a similar behavior for *grid*. As we stated earlier, the behavior is attributable to the misalignment in orientation between the input and the normal images. Our method is free of this misalignment issue, leading to better performance.

### 5.4. Results on DTD-Synthetic

For further verification, we test our method and others on DTD-Synthetic. (We choose PatchCore for comparison.) Table 3 shows the detection accuracy in the image-level and pixel-level AUROC. Figure 3 shows example pairs of an input image and its anomaly score map for the twelve textures. The method achieves over 97.2% image-level AUROC for all 12 textures and 100% for 5 textures.

We then compare our zero-shot method with SPADE and PatchCore. To test their performanace in few-shot settings, we change the number of normal sample images from one to 100. Figure 4 show PatchCore's image-level AUROC vs. the number of available normal images for the three texture categories. Note that ours is a zero-shot method and thus its accuracy is constant with the horizontal axis.

Overall, our method performs better than PatchCore even with 100 normal images; specifically, the average image-level AUROC over the twelve textures is 98.9% (ours) vs. 98.0% (PatchCore with 100-shots). More detailed observation is as follows.

First, the Category-1 textures are the most difficult in the three categories for the both methods. Our method attains AUROC between 97.2 and 98.9%. This is much better than PatchCore in few($\leq 5$)-shot settings, whose accuracy is lower than 80% except for *perforated_037*. PatchCore with 100 normal images is still worse than our zero-shot method for three out of four textures.

We can see that PatchCore's accuracy deteriorates with the number of shots, similar to *grid* in MVTec AD. As with *grid*, the textures in this category are all anisotropic and their images have random orientation. These results agree well with our argument of texture orientation in Sec. 3.

The proposed method achieves high accuracy ($> 98.6\%$) also for the Category-2 textures. PatchCore is worse than ours in the few-shot settings; its accuracy is low ($< 90\%$) especially for *woven_068* and *woven_104*. As in Category-1, PatchCore with 100 normal images is also worse than our zero-shot method except for *woven_125* in which both method produce AUROC of 100%. As with Category-1, the textures in this category are anisotropic and have random orientations.

On the other hand, the proposed method and PatchCore both attain high accuracy for the Category-3 textures. (Note the range of the vertical axis of the plot.) It is noteworthy that PatchCore attains 100% accuracy in the 1-shot setting for *blotcy_099*, which is the same as our method. These results further verify our argument. The textures in this category are isotropic (or do not show clear anisotropy). Thus, texture orientation does not matter since there is no orientation. These results further support our argument.

An additional remark is that our method shows worse performance for *fiborous_183* than other three textures, while it is still better than PatchCore in few-shot settings. This will be because of the relatively higher irregularity of the texture. As seen from the example in Fig. 3, *fiborous_183* is less homogeneous from others. We will discuss the applicability of our method.

Table 2. Performance (image-level AUROC) on five textures of MVTec AD of our proposed zero-shot method, SPADE [6], PatchCore[17], and RegAD[12] with different numbers of normal images. † indicates the accuracy reported in the papers.

| Method | Ours | SPADE | | | | | | PatchCore | | | | | | RegAD | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Num. of Shots | 0 | 1 | 5 | 10 | 100 | All | All† | 1 | 5 | 10 | 100 | All | All† | 2† | 4† | 8† |
| **average** | **99.6** | **79.0** | **79.8** | **81.0** | **83.2** | **85.4** | - | **89.5** | **93.0** | **95.7** | **98.5** | **98.8** | **99.0** | **94.7** | **96.6** | **97.4** |
| carpet | 99.9 | 82.6 | 85.7 | 85.5 | 86.7 | 87.3 | - | 98.6 | 98.7 | 98.6 | 98.8 | 98.6 | 98.7 | 96.5 | 97.9 | 98.5 |
| grid | 100 | 40.9 | 43.5 | 45.2 | 48.8 | 56.9 | - | 52.3 | 69.5 | 82.7 | 96.6 | 97.3 | 98.2 | 84.0 | 91.2 | 91.5 |
| leather | 100 | 86.3 | 83.1 | 86.3 | 88.6 | 90.3 | - | 100 | 100 | 100 | 100 | 100 | 100 | 99.4 | 100 | 100 |
| tile | 99.1 | 91.0 | 93.4 | 92.4 | 94.9 | 95.5 | - | 98.4 | 98.5 | 98.6 | 98.6 | 98.9 | 98.7 | 94.3 | 95.5 | 97.4 |
| wood | 98.9 | 94.1 | 93.2 | 95.4 | 97.2 | 97.0 | - | 98.3 | 98.6 | 98.6 | 98.8 | 99.4 | 99.2 | 99.2 | 98.6 | 99.4 |

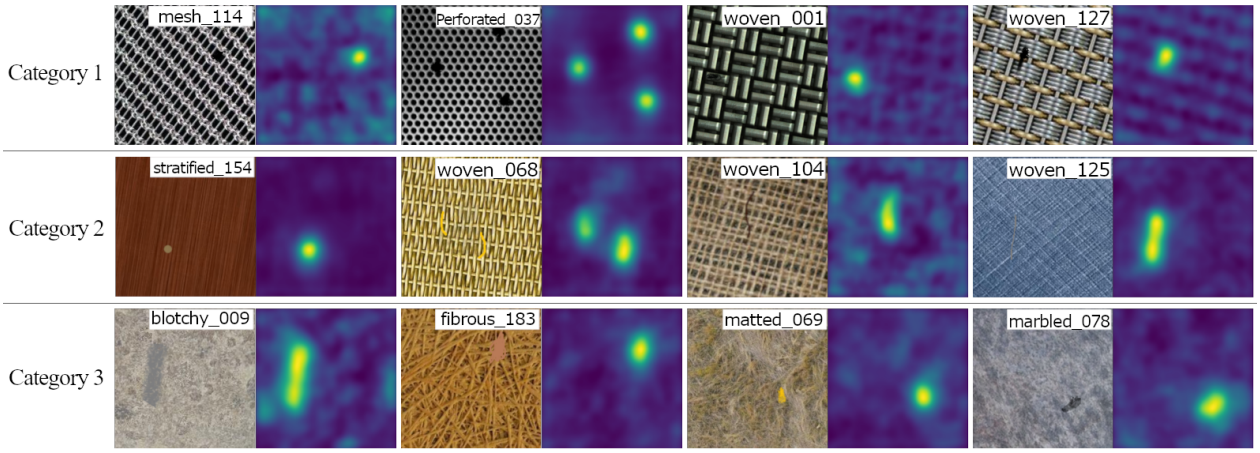


Figure 3. Example results of the proposed zero-shot method on the DTD-Synthetic dataset. Each pair shows the input image and the anomaly score map.

Table 3. Detection accuracy (AUROC) of the proposed zero-shot method on the DTD-Synthetic dataset.

| Cat. | Class | Image AUROC | Pixel AUROC |
|---|---|---|---|
| | **All class** | **98.9** | **98.0** |
| | **Cat. average** | **98.0** | **96.8** |
| | Mesh_114 | 98.0 | 97.3 |
| 1 | Perforated_037 | 98.9 | 97.4 |
| | Woven_001 | 97.9 | 99.5 |
| | Woven_127 | 97.2 | 93.0 |
| | **Cat. average** | **99.3** | **98.4** |
| | Stratified_154 | 100 | 99.2 |
| 2 | Woven_068 | 98.6 | 98.3 |
| | Woven_104 | 98.6 | 97.1 |
| | Woven_125 | 100 | 98.9 |
| | **Cat. average** | **99.5** | **98.9** |
| | Blotchy_099 | 100 | 99.2 |
| 3 | Fibrous_183 | 97.8 | 98.3 |
| | Matted_069 | 100.0 | 99.3 |
| | Marbled_078 | 100.0 | 98.9 |

## 5.5. Predicting the Method's Applicability

As explained in Sec. 4.3, we propose to use $\alpha(I)$ of Eq. (3) for predicting the applicability of the proposed zero-shot method for an anomaly-free image $I$ of the target texture. To test its effectiveness, we apply the method to the images of MVTec AD and DTD. For MVTec AD, we choose the first image from the training splits of each of all the categories, including five textures and ten objects. For DTD, we use the twelve textures comprising DTD-Synthetic and additional textures from the rest of DTD textures.

Figure 5 shows the quantity $\alpha(I)$ vs. the image-level AUROC for each texture. It is observed that there exists fairly strong correlation between the two axes. The accuracy is very high for all the five textures of MVTec AD, as is reported above, whereas it is very low for all the ten object textures. For DTD textures, $\alpha(I)$ distributes in a wide range, which well match the spread of their detection accuracy. We show some of the DTD textures for which the accuracy is low and $\alpha(I)$ is high in the supplementary material. Although they are called textures, they have considerably different brightness structures from those considered in anomaly detection scenario. We can also observe from Fig. 5 that the detection accuracy is close to 100% for the textures with $\alpha(I)$ lower than 1.5. Thus, it will be reasonable to use the value for the threshold to judge the method's applicability.

## 5.6. Results on Other Datasets

We show more results on other datasets to demonstrate how the above strategy works. We choose Aitex fabrics images [21] and DAGM2007 [22] here. The former consist of 245 images of seven types of fabric. Following previous studies, we split each image with a long horizontal into multiple images and then resize them into $320 \times 320$. We use all the textures since they satisfy the criterion $\alpha(I) \leq 1.5$. The latter (i.e., DAGM2007) contains ten classes of texture im-

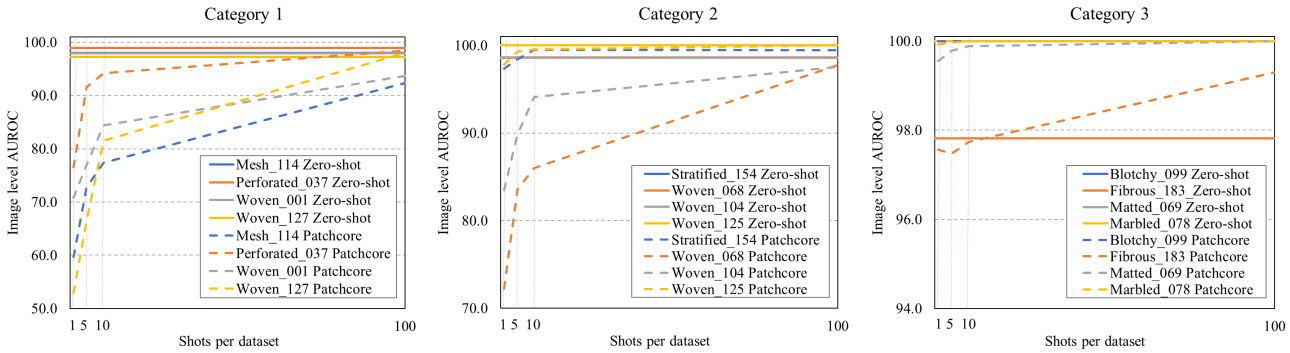


Figure 4. Detection accuracy (image-level AUROC) on DTD-Synthetic by the proposed method (solid lines) and PatchCore (dotted lines) with different numbers $N$ of normal images. As our method does not use a normal image, accuracy does not change with $N$. Some results from the proposed method (solid lines) are overlapping with each other as the AUROC achieved is close to 100%.

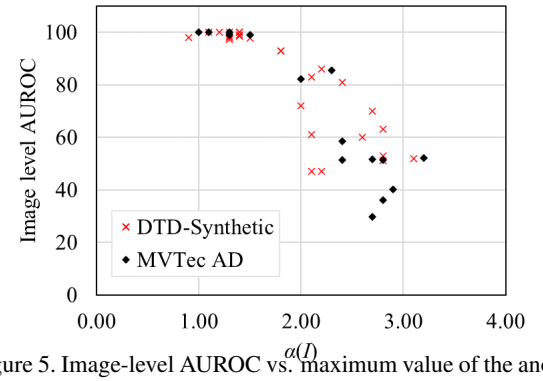


Figure 5. Image-level AUROC vs. maximum value of the anomaly score $\alpha(I)$ on MVTec AD and DTD-Synthetic datasets.

ages with artificially synthesized anomalies. We resize the original image with $512 \times 512$ into $320 \times 320$. We choose seven textures that satisfy the criterion; it is noted that $\alpha(I)$ is around 1.5 for six of them. Table 4 shows the results for Aitex and 5 shows those for DAGM2007. Our zero-shot method works reasonably well; it outperforms PatchCore in few-shot settings for Aitex and in 100-shot for DAGM2007.

Table 4. Results (Image level AUROC (%))) on Aitex.

| Methods | 00 | 01 | 02 | 03 | 04 | 05 | 06 |
|---|---|---|---|---|---|---|---|
| Ours | 93.9 | 100 | 96.3 | 96.1 | 100 | 100 | 100 |
| PC-1shot | 90.2 | 100 | 85.4 | 91.7 | 93.3 | 100 | 97.7 |
| PC-5shot | 91.7 | 100 | 91.3 | 92.6 | 97.8 | 100 | 99.4 |

Table 5. Results (Image level AUROC (%))) on DAGM2007.

| Methods | C2 | C3 | C5 | C6 | C7 | C8 | C9 |
|---|---|---|---|---|---|---|---|
| Ours | 99.9 | 96.2 | 95.7 | 98.2 | 98.9 | 79.4 | 97.1 |
| PC-100shot | 97.1 | 88.7 | 89.4 | 96.9 | 95.3 | 78.4 | 84.5 |

## 6. Summary and Conclusion

We have considered unsupervised anomaly detection for texture images. We first pointed out a crucial nature of the problem that has been overlooked in the literature: accurate detection gets harder for anisotropic textures when image orientations are unaligned. We then proposed a zero-shot method for the problem. As it does not need a normal sample, it is free from the orientation issue. The proposed method assumes the input textures to be homogeneous, detecting image regions that break the homogeneity as anomalies. We have proposed a quantitative criterion to judge whether the assumption holds for input textures, which enables to predict the proposed method effectiveness for each texture. Experimental results show the effectiveness of the proposed approach. Specifically, the zero-shot method attains an average image-level AUROC of 99.6% over five textures of MVTec AD, which is better than PatchCore (99.0%), the current state-of-the-art method. Note that PatchCore yields the best result when using hundreds of normal images; its accuracy deteriorates in few-shot settings, due to the above orientation issue with anisotropic textures. For more detailed analyses, we created a synthetic dataset, named DTD-Synthetic, using the texture images from DTD (Describable Texture Dataset) and adding synthetic anomalies simulating natural texture defects. The results using the dataset shows that the proposed method outperforms PatchCore even in many-shot settings for anisotropic textures and shows comparable results for isotropic textures.

**Acknowledgments:** This work was partly supported by JSPS KAKENHI Grant Number 20H05952 and 19H01110.

## References

[1] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. MVTec AD–a comprehensive real-world dataset for unsupervised anomaly detection. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[2] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Uninformed students: Student-teacher

anomaly detection with discriminative latent embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4183–4192, 2020.

[3] Paul Bergmann, Sindy Löwe, Michael Fauser, David Sattlegger, and Carsten Steger. Improving unsupervised defect segmentation by applying structural similarity to autoencoders. In *Proc. International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP)*, 2019.

[4] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 2009.

[5] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proc. the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[6] Niv Cohen and Yedid Hoshen. Sub-image anomaly detection with deep pyramid correspondences. *CoRR*, 2020.

[7] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. Padim: A patch distribution modeling framework for anomaly detection and localization. In *Proc. International Conference on Pattern Recognition (ICPR)*, 2021.

[8] David Dehaene, Oriel Frigo, Sébastien Combrexelle, and Pierre Eline. Iterative energy-based projection on a normal data manifold for anomaly localization. In *Proc. International Conference on Learning Representations (ICLR)*, 2020.

[9] Hanqiu Deng and Xingyu Li. Anomaly detection via reverse distillation from one-class embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9737–9746, 2022.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of International Conference on Learning Repreesntation*, 2021.

[11] Denis Gudovskiy, Shun Ishizaka, and Kazuki Kozuka. Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 98–107, 2022.

[12] Chaoqin Huang, Haoyan Guan, Aofan Jiang, Ya Zhang, Michael Spratling, and Yan-Feng Wang. Registration based few-shot anomaly detection. In *European Conference on Computer Vision (ECCV)*, 2022.

[13] 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.

[14] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. Deep learning for anomaly detection: A review. *ACM Computing Surveys (CSUR)*, 2021.

[15] Marco AF Pimentel, David A Clifton, Lei Clifton, and Lionel Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.

[16] Oliver Rippel, Patrick Mertens, and Dorit Merhof. Modeling the distribution of normal data in pre-trained deep features for anomaly detection. In *Proc. International Conference on Pattern Recognition (ICPR)*, 2021.

[17] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14318–14328, 2022.

[18] Marco Rudolph, Tom Wehrbein, Bodo Rosenhahn, and Bastian Wandt. Fully convolutional cross-scale-flows for image-based defect detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1088–1097, 2022.

[19] 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.

[20] 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 *Proc. International Conference on Information Processing in Medical Imaging (ICIPMI)*, 2017.

[21] Javier Silvestre-Blanes, Teresa Albero Albero, Ignacio Miralles, Rubén Pérez-Llorens, and Jorge Moreno. A public fabric database for defect detection methods and results. *Autex Research Journal*, 19(4):363–374, 2019.

[22] Matthias Wieler and Tobias Hahn. Weakly supervised learning for industrial optical inspection. In *DAGM symposium in*, 2007.

[23] Jiawei Yu, Ye Zheng, Xiang Wang, Wei Li, Yushuang Wu, Rui Zhao, and Liwei Wu. Fastflow: Unsupervised anomaly detection and localization via 2d normalizing flows. *arXiv preprint arXiv:2111.07677*, 2021.

[24] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proc. British Machine Vision Conference (BMVC)*, 2016.