Self-Supervised Normalizing Flows for Image Anomaly Detection and Localization

Li-Ling Chiu, Shang-Hong Lai
Department of Computer Science, National Tsing Hua University, Taiwan
clairechiu1997@gmail.com, lai@cs.nthu.edu.tw

Abstract

Image anomaly detection aims to detect out-of-distribution instances. Most existing methods treat anomaly detection as an unsupervised task because anomalous training data and labels are usually scarce or unavailable. Recently, image synthesis has been used to generate anomalous samples which deviate from normal sample distribution for model training. By using the synthesized anomalous training samples, we present a novel self-supervised normalizing flow-based density estimation model, which is trained by maximizing the likelihood of normal images and minimizing the likelihood of synthetic anomalous images. By adding constraints to abnormal samples in our loss function, our model training is focused on normal samples rather than synthetic samples. Moreover, we improve the transformation subnet of the affine coupling layers in our flow-based model by dynamic stacking convolution and self-attention blocks. We evaluate our method on MVTec-AD, BTAD, and DAGM datasets and achieve state-of-the-art performance compared to flow-based and self-supervised methods on both anomaly detection and localization tasks.

1. Introduction

Anomaly detection aims to detect samples that are obviously distinct from normal patterns. It is a trending topic in computer vision with diverse applications, including industrial image defect detection, medical diagnostics, video surveillance, etc. Nevertheless, anomaly detection is often posed as a one-class classification problem because in many cases only normal data is available for training. Moreover, the scarcity and diversity of anomalous samples make the collection of complete defective samples infeasible.

Most of the current anomaly detection approaches are unsupervised methods [5, 6, 8, 27, 30, 35, 41, 43], which are trained only on non-defect images. Those works are based on generative models such as Adversarial Generative Networks (GANs) [1, 2, 20, 34, 35], Variational Autoencoders (VAEs) [6, 25, 45, 46], or normalizing flows (NFLOWs) [13, 28, 29, 44]. After learning the probability distribution of normal data, the out-of-distribution testing images are detected as abnormal from the model inference. However, model training without anomalous images faces challenges of detecting defective images that are slightly different from training samples due to the lack of knowledge on the anomaly.

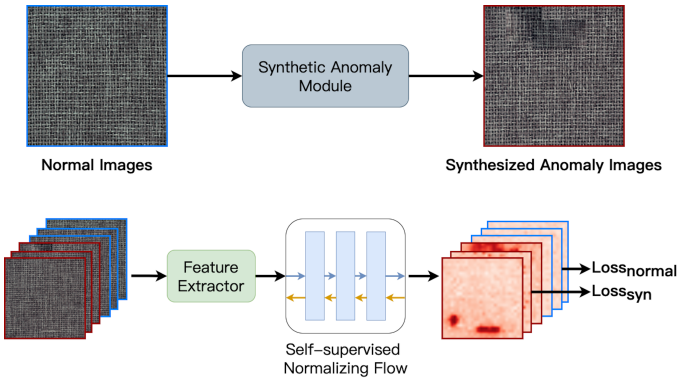

Figure 1. Overview of our self-supervised normalizing flow-based model. First, we generate artificial defective images using a synthetic anomaly module Second, we use a pre-trained model as our feature extractor for both artificial defective images and original normal images. Finally, we calculate $Loss_{syn}$ and $Loss_{normal}$ based on density estimation and train our model to learn the distribution of normal features.

We can reduce the imbalance between normal and abnormal images for model training using synthetic images. Some self-supervised approaches have recently utilized data augmentation with traditional image processing methods to produce synthetic anomalous data. After learning more information about potential anomalous regions, these self-supervised methods have shown to be successful in distinguishing normal data from outliers.

In order to leverage the benefit of self-supervised methods and normalizing flow-based models, we propose a self-

supervised normalizing flow-based model for anomaly detection and segmentation, illustrated in Fig. 1. Our model is trained on a set of real normal and synthetic abnormal images. By employing a conditional loss on the abnormal images, we can improve the density estimation of normal feature with the proposed flow-based model.
The main contributions of our paper are summarized as follows:

- We present a self-supervised normalizing flow-based model that includes synthetic abnormal data into training to improve the model accuracy on anomaly detection.
- We propose a conditional loss function in conjunction with a stable training process to prevent our model from being significantly influenced by extreme abnormal samples.
- We propose a dynamic transformation network by allowing the coupling layers to tune different learnable layers for our flow-based model.
- Our self-supervised learning method achieves state-of-the-art performance on several public anomaly detection benchmarks.

2. Related Work

Most existing unsupervised anomaly detection methods assume that only normal data is available during the training. Some generative methods, such as autoencoder-based [6,25,45,46] methods and GAN-based methods [1,2, 20,34,35], focus on image reconstruction and detect anomalies based on image reconstruction errors. Anomalous regions can be spotted as they are not well reconstructed by the model trained on normal data only. However, these methods face challenges in that their models reconstruct well on normal and abnormal samples due to the generalization of CNN models.

Another generative model, normalizing flows (NFs) have been successfully used for anomaly detection [13, 28, 29, 44]. NFs models trained on normal samples and learn the distribution. [29] proposed a multi-scale flow to obtain representation from different scales of images. [13] used conditional normalizing flows for multi-scale feature, and proposed a new scoring function for anomaly localization. After model training, the likelihood of individual images can be considered as the anomaly score.

A recent emerging direction focuses on self-supervised learning. One major family is based on reconstructing non-defect images from generated defective images [25, 45, 46]. [45] trained autoencoder-based models to reconstruct normal images and trained another model for calculating an anomaly score on every pixel. Some methods treat augmented images as negative samples and train pixel-wise [36] or image-based [9, 19, 24, 38] classification models. On the other hand, [26] treats failure cases from the training process of the generator as out-of-distribution samples. [16] utilizes an adversarial training strategy between their random mask model and reconstruction model to learn a feature representation with semantic information.

Although anomalous types are unpredictable, a small number of (e.g., one to multiple) labeled anomaly examples are often available in many relevant real-world applications. Some supervised methods tried to learn feature information on both anomalous and normal images. With the additional knowledge of application-specific abnormality, supervised methods can detect samples that are slightly different from normal images. [24] used a small set of real abnormal images and utilized one-sided anomaly deviation loss for model training on an imbalanced dataset. A multi-task classification method is introduced in [9] to classify normal data, real anomalous samples, and synthetic anomalies. [37] fine-tuned their model with outliers to fail on reconstructing out-of-distribution samples.

3. Proposed Method

To detect anomalous images, we aim to learn the distribution of normal samples. Given N normal images $U = \{u_1, u_2, \cdots, u_N\}$, we apply the NSA method [36] to generate N synthetic defect images $S = \{s_1, s_2, \cdots, s_N\}$. After that, we combine normal images and synthetic images into our training dataset $M = \{m_1, m_2, \cdots, m_N, m_{N+1}, m_{N+2}, \cdots, m_{2N}\}$. The first N samples are normal images, and the remaining N samples are synthetic images.

The architecture and training pipeline of our proposed model is illustrated in Fig. 3. We first use a deep model pretrained on ImageNet [32] as our feature extractor $f_{fe} : M \rightarrow X$. f_{fe} extracts representation features x_i for every training image m_i while the weights remain unchanged during the model training.

After that, we train our normalizing flow-based model to estimate the distribution of normal features. Our model is also optimized to distinguish between in-distribution and out-of-distribution data with the information from our synthetic anomaly samples and self-supervised loss function, which will be detailed in the subsequent subsections.

3.1. Self-Supervised Learning

In some self-supervised methods, their models learn deeper representations of normal images by artificially generating abnormal images. Existing self-supervised methods, such as CutPaste [19], FPI [39], NSA [36] and DRAEM [45], all employed some image synthesis methods for generating anomalous images. To optimize our model for distinguishing normal and anomalous samples during

the model training, our training dataset contains normal samples and synthetic anomalous samples. In this work, we follow Poisson Image Editing proposed in [36] as our image synthesis mechanism to generate anomalous images that are close to real-world situations.

Figure 2 illustrates this synthetic anomaly generation module. Given two normal images u_{src} and u_{dest}, we cut patches from u_{src} and blend them on u_{dest} image. We first sample the width and height from a truncated Gamma distribution and randomly resize the patch. Lastly, we use the blending algorithm to blend the patch on u_{dest} at a random place. We repeat the steps above to paste a random number of patches on u_{dest} to generate a synthetic anomalous image. Besides, this method uses the brightness of object images to produce an object mask for every object image to avoid pasting the patches on the background. By creating object mask m_{src} and m_{dest} for u_{src} and u_{dest}, we can ensure that the patch contains object parts and confirm that each patch is attached to the object.

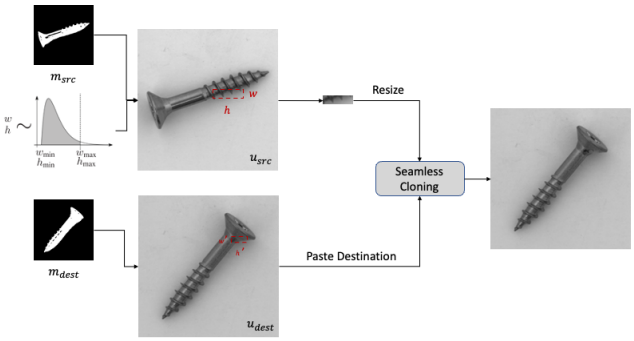

Figure 2. **Synthetic Anomaly Module.** The image synthesis process includes sampling width and height from a gamma distribution, producing object masks to avoid background for source and destination images, and blending patches at random places in the destination image.

3.2. Normalizing Flows Architecture

Normalizing flows [10, 11, 17] are trained to estimate the likelihood of the training set directly. Their invertible transformation function bijectively maps image distribution $p_X(x)$ into a latent space distribution $p_Z(z)$. The likelihood for arbitrary data distribution can be formulated as Eq. 1 by utilizing the change of variables formula.

$$\log p_X(x) = \log p_Z(f_\theta(x)) + \log \left| \det \frac{\partial f_\theta(x)}{\partial x} \right| \quad (1)$$

The latent space distribution is often modeled as a standard Gaussian. Normalizing flow-based models are trained to maximize the log-likelihood of training data using their transformation function $f_\theta(x)$.

We use the affine coupling layer in [10, 11] to form our flow-based model architecture to efficiently calculate the second part in Eq. 1. In one affine coupling layer, the input x is randomly permuted and split across the channel dimension into two parts, x_1 and x_2. The output y is concatenated by y_1 and y_2 along channel dimension. We illustrate this structure in Fig. 3. The transformation in each layer follows the following equations

$$y_1 = x_1, \quad (2)$$
$$y_2 = x_2 \odot \exp(s(x_1)) + t(x_1), \quad (3)$$

where $\odot$ is the element-wise multiplication operation. Two operation functions $s(\cdot)$ and $t(\cdot)$ are the output of one subnetwork, which will be described in Section 3.2.1. Therefore, $\log \left| \det \frac{\partial f_\theta(x)}{\partial x} \right|$ can be calculated by the Jacobian determinant of this coupling layer. We stack N multiple coupling layers in our model to enhance model complexity, thus making our model capable of learning more complicated distribution for normal samples. The Jacobian matrix of the flow-based model can be computed by multiplying the Jacobian matrix of each coupling layer.

3.2.1 Residual Connected Subnet

We propose a dynamic transformation subnet of our affine coupling layers, illustrated in Fig. 3. In Eq. 3, $s(\cdot)$ and $t(\cdot)$ are implemented by one neural network, i.e. the subnet of coupling layers. In order to learn both local and global information, our subnet is combined with 1*1 and 3*3 convolution layers and one 4-head self-attention layer. Besides, to dynamically tune different learnable layers, we use the residual connection to contain the output of the former layer and the current layer, similar to [14, 15]. With residual connections, we can consider that our subnets have different branches of interior layers. Therefore, our subnet can integrate the advantages of convolution layers and multi-head self-attention layers. They can also adjust the number of layers during training time to learn an optimistic subnet architecture for different datasets and classes. The multi-head self-attention layer is identical to the one in the Transformer [42]. In Section 5.2, we discuss the influence of the combination of different layers and the residual component.

3.3. Learning Objective

With the additional synthetic abnormal images, we can extend the unsupervised normalizing flows in Section 3.2 to the self-supervised task. We first use a pretrained model to extract normal and anomalous image features from our training dataset T. Since normalizing flow-based models are first designed to learn the distribution of given data, we focus on learning the distribution of normal samples here.

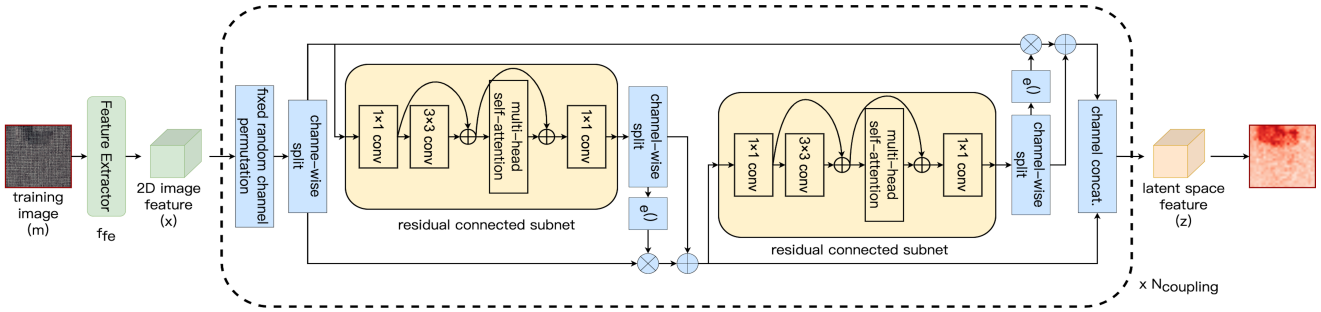

Figure 3. The architecture of our whole model. We illustrate one block inside our normalizing flow-based model. The input x is the extracted feature of our training image m. Our model bijectively maps the image feature distribution into the latent space, modeled as a Gaussian distribution.

Our model is optimized with two goals: 1. learn the distribution of normal samples and 2. refine the distribution by employing abnormal samples as out-of-distribution constraints.

Given a set of N real normal features $D = \{x_d\}_{d=1}^{N}$ and a set of N synthetic image features $S = \{x_s\}_{s=1}^{N}$, we train our density estimation model to maximize the likelihood of normal samples and minimize the likelihood of our synthesized abnormal samples. We optimize our flow model f_θ with the following objective function:

$$\arg\max_{\theta} \frac{1}{N}\sum_{x_d \in D} \log p_D(x_d) - \frac{1}{N}\sum_{x_s \in S} \log p_S(x_s) \quad (4)$$

We reformulate our objective function to train our model to minimize the negative log-likelihood for normal samples, $-\log p_D(x_d)$, and maximize the negative log-likelihood for synthetic anomaly samples, $-\log p_S(x_s)$. Therefore, following Eq. 1, we calculate the negative log-likelihood for normal samples by

$$\begin{aligned} & L_{normal} \\ &= \frac{1}{N}\sum_{x_d \in D} -\log p_D(x_d) \\ &= \frac{1}{N}\sum_{x_d \in D} \left[-\log p_Z(f_\theta(x_d)) - \log \left| \det \frac{\partial f_\theta(x_d)}{\partial x_d} \right| \right]. \end{aligned} \quad (5)$$

Inspired by [12, 18], our loss function for synthetic anomalous samples is defined as

$$\begin{aligned} L_{syn} = \frac{1}{N}\sum_{x_s \in S} & -\log p_S(x_s) \cdot I[\log p_S(x_s) > c \\ & \wedge \log p_S(x_s) > min(\log p_D(x_d))], \end{aligned} \quad (6)$$

where c is a constant threshold and $I[\cdot]$ is an indicator function. We only include synthetic samples satisfying the condition in $I[\cdot]$ to compute our synthetic distribution. In order to prevent the log-likelihood of synthesized samples $\log p_S(x_s)$ from reaching to $-\infty$, we follow the setting in [18] to encourage the flow to push the synthetic log-likelihood at least the threshold c. Because our normal images and synthetic images are slightly different, our second condition, inspired by [12], removed the synthetic samples fulfilling our training optimization (the likelihood of all normal samples should be larger than synthetic samples) from the loss function.

The final loss L_{total} for training our self-supervised flow-based model is given by Eq. 7, which is designed to prioritize decreasing the negative log likelihood for normal images.

$$L_{total} = L_{normal} - L_{syn} \quad (7)$$

3.4. Scoring Function

After the training, we follow other 2D flow-based methods [29, 44] and use the calculated likelihood of testing samples as the classification criteria between normal and anomalous samples. Given a testing image p, we use feature extractor f_{fe} and our self-supervised flow f_θ to calculate the likelihood $z \in R^{H \times W \times C}$. Samples with lower likelihood would be considered anomalous. Therefore, we upsample the negative likelihood $-z$ and aggregate the values along channel-wise dimension as our anomaly segmentation score. Otherwise, the detection anomaly score for every image is the maximum value of its segmentation map.

4. Experimental Results

We perform our experiments on **MVTec-AD** [4], **BTAD** [23], and **DGAM** [22] datasets. These datasets are popular benchmarks for image anomaly detection. The performance of our model and all related methods included in our comparison is evaluated with the Area Under the Receiver Operating Characteristic Curve (AUROC) % at image-level and pixel-level, with numbers in **Bold** representing the best results in the comparison and the underlined numbers representing the best result among all self-supervised methods.

4.1. Experimental Comparisons

In these experiments, we choose CaiT [40], a transformer model proposed by Facebook AI Team, as our pretrained model. Our flow-based model is implemented based on [3] framework. We compared our method with several state-of-the-art methods. DifferNet [28], CFLOW-AD [29], and CS-Flow [29] are unsupervised normalizing flow-based models. DRAEM [45], CutPaste [19], and NSA [36] are self-supervised methods. Note that we used unofficial implementation from [31] for BTAD experiments.

4.1.1 Results on MVTec AD

Table 1 and Table 2 show the image-level and pixel-level anomaly detection experimental results on MVTec-AD. Our model achieves a 100% AUROC score in eight classes. The average score of 15 classes gives the highest 98.7% AUROC score among these methods. Moreover, our model outperforms other methods in texture classes, which reach a 99.9% average score. Among all self-supervised methods, our method outperforms all competitors in 13 classes and reaches state-of-the-art results. With the additional information on potential defective images, we exceed some challenging classes, such as hazelnut and metal_nut. Different from CutPaste [19], we use brightness to avoid the background of object classes in our synthetic mechanism. We not only obtain a higher score on texture classes but also highly exceed CutPaste on object classes with the help of natural synthetic anomaly images. Moreover, unlike other self-supervised classification models, our training loss function focused on learning the distribution of normal samples. Our models can detect anomaly types different from our synthesized training samples and they are not easily influenced by artificial defect types. Besides, our model exceeds CS-Flow, the other 2D normalizing flow-based method, in pill and capsule classes. Those classes have relatively small objects, and some models tend to detect small defects in the background. In addition, although our loss function was designed to optimize the model for the anomaly detection task, our AUROC score exceeds 95% in every class and gets a 98.1 average score in the anomaly segmentation task.

4.1.2 Results on BTAD Dataset

Table 3 and Table 4 show our experimental comparison results on BTAD dataset. Our model outperforms other competitive methods on Product 02. By sampling the width and height from a truncated gamma distribution, the synthesizing mechanism tends to create defective images with small defective areas. Our model can learn to compute the lower likelihood of images slightly different from normal samples. However, unlike CFLOW-AD [13] trained three flow-based models with different image sizes, we trained with only one complex flow model. On the other hand, our loss function focuses on the identification of anomaly images. Therefore, despite the fact that our model is capable of detecting images with slight defeats, our pixel-level segmentation results on minor scratches on Product 02 are not accurate enough.

Table 1. Image-level anomaly detection comparison results on MVTec-AD trained on full-dataset.

	Unsupervised Methods			Self-Supervised Methods			
	[28]	[13]	[29]	[45]	[19]	[36]	Ours
carpet	84.0	**100**	**100**	97.0	93.9	95.6	99.7
grid	97.1	97.6	99.0	99.9	**100**	99.9	**100**
leather	99.4	97.7	**100**	**100**	**100**	99.9	**100**
tile	92.9	98.7	**100**	99.6	94.6	**100**	**100**
wood	99.8	99.6	**100**	99.1	99.1	97.5	**100**
Avg.Texture	94.6	98.7	99.8	99.1	97.5	98.6	**99.9**
bottle	99.0	**100**	99.8	99.2	98.2	97.7	**100**
cable	86.9	**100**	99.1	91.8	81.2	94.5	98.1
capsule	88.8	99.3	97.1	98.5	98.2	95.2	97.6
hazelnut	99.1	96.8	99.6	**100**	98.3	94.7	**100**
metal_nut	95.1	91.9	99.1	98.7	99.9	98.7	**100**
pill	95.9	**99.9**	98.6	98.9	94.9	99.2	99.6
screw	99.3	**99.7**	97.6	93.9	88.7	90.2	94.1
toothbrush	96.1	95.2	91.9	**100**	99.4	**100**	92.2
transistor	96.3	99.1	**99.3**	93.1	96.1	95.1	99.1
zipper	98.6	98.5	99.7	**100**	99.9	99.8	**100**
Avg. Object	95.5	98.0	**98.2**	97.4	95.5	96.5	98.1
Avg. All	94.9	98.3	**98.7**	98.0	96.1	97.2	**98.7**

Table 2. Pixel-level anomaly segmentation comparison results on MVTec-AD trained on full-dataset.

	CFLOW [13]	DRAEM [45]	CutPaste [19]	NSA [36]	Ours
carpet	**99.3**	95.5	98.3	95.5	**99.3**
grid	99.0	**99.7**	97.5	99.2	98.3
leather	**99.7**	98.6	99.5	99.5	99.5
tile	98.0	99.2	90.5	**99.3**	96.5
wood	**96.7**	96.4	95.5	90.7	95.4
Avg. Texture	**98.5**	97.9	96.3	96.8	97.8
bottle	99.0	99.1	97.6	98.3	98.1
cable	97.6	94.7	90.0	96.0	**97.7**
capsule	**99.0**	94.3	97.4	97.6	98.6
hazelnut	98.9	99.7	97.3	97.6	99.1
metal_nut	98.6	99.5	93.1	98.4	98.2
pill	**99.0**	97.6	95.7	98.5	98.9
screw	**98.9**	97.6	96.7	96.5	**98.9**
toothbrush	**99.0**	98.1	98.1	94.9	98.6
transistor	**98.0**	96.4	93.0	88.0	95.6
zipper	99.1	98.8	**99.3**	94.2	99.0
Avg. Object	**98.7**	97.6	95.8	96.0	98.3
Avg. All	**98.6**	97.3	96.0	96.3	98.1

Table 3. Image-level anomaly detection comparison results on BTAD [23] full dataset.

	Unsupervised Methods			Self-Supervised Methods			
	[28]	[13]	[29]	[45]	[19]	[36]	Ours
Product 1	99.1	97.4	99.4	98.6	99.8	**100**	99.2
Product 2	85.4	85.9	87.5	78.1	87.1	84.7	**92.2**
Product 3	98.5	99.4	**100**	98.8	**100**	99.0	98.3
Average	94.3	94.2	95.6	91.9	95.6	94.6	**96.6**

Table 4. Pixel-level anomaly segmentation comparison results on BTAD [23] full dataset.

	CFLOW [13]	DRAEM [45]	NSA [36]	Ours
Product 1	94.7	78.6	96.7	**96.9**
Product 2	**96.8**	75.4	88.9	92.8
Product 3	**99.6**	66.3	99.5	99.5
Average	**97.0**	73.4	95.0	96.4

4.1.3 Results on DAGM

DGAM [22] is a synthetic anomaly detection dataset focused on texture surfaces. Most of the existing methods compare their results on the image-level anomaly detection task because DGAM only provides weak segmentation labels, roughly indicating the defective area using ellipses.

Table 5 shows our image-level comparison results. [21] and [7] are self-supervised methods, which were trained on the whole training dataset, including both normal and defective images with segmentation labels. Unsupervised methods [2, 20, 28] and self-supervised methods [19, 33, 45] skip all labeled anomaly training data. The results show that our model significantly exceeds all unsupervised and self-supervised competitors. We reach a 100% AUROC score in nine classes and have a 100% average score. Moreover, We achieve the performance of those supervised methods, but our model is trained on normal images only.

4.1.4 Few-shot Experiment

We extend the training setting to a few-shot learning area on MVTec-AD [4] and BTAD [23] to ensure the model's robustness. In our few-shot scenario, we trained models on 16 random normal images and evaluated them on the full testing dataset. Table 6 shows the detection results on MVTec, and Table 7 shows the segmentation results on BTAD. The experimental results demonstrate that our model achieves high anomaly detection and segmentation results with limited training data and our model significantly outperforms the other methods.

By stacking different learning layers in our subnet, our model is capable of learning diverse features. Moreover, our self-supervised conditional loss function prevents our model from focusing on abnormal features, so optimization to learn the distribution of normal features prevails over minimizing the likelihood of abnormal features. Therefore, our model remains high accuracy on some challenging object classes, such as pill, hazelnut, and screw, with limited training samples. The few-shot experiments prove that the performance of our model remains stable for both full-shot and few-shot scenarios.

Table 5. Image-level anomaly detection comparison results on full-dataset. We use C, SL, USL, and SSL abbreviations for class, supervised, unsupervised, and self-supervised learning, respectively.

	SL		USL			SSL			
	[21]	[7]	[2]	[20]	[28]	[33]	[19]	[45]	Ours
C 1	**100**	**100**	58.3	99.1	59.7	50.7	56.1	96.1	99.6
C 2	94.0	**100**	56.1	**100**	82.9	50.5	87.8	98.3	**100**
C 3	**100**	**100**	55.1	99.1	69.8	58.7	57.1	99.5	**100**
C 4	**100**	**100**	53.7	99.0	97.3	70.0	71.3	99.6	**100**
C 5	**100**	99.9	57.4	**100**	61.2	63.6	47.4	92.1	**100**
C 6	**100**	**100**	66.8	97.5	97.0	92.3	68.8	**100**	**100**
C 7	**100**	**100**	52.4	99.8	68.5	54.0	96.5	99.7	**100**
C 8	99.0	**100**	53.7	99.8	52.1	49.1	53.4	99.9	**100**
C 9	**100**	**100**	52.3	99.5	78.2	54.6	51.9	98.9	**100**
C 10	**100**	**100**	52.2	99.2	79.1	49.6	74.7	96.0	**100**
Avg.	99.3	**100**	55.8	99.3	74.6	59.3	66.0	98.0	**100**

Table 6. 16 shot image-level anomaly detection comparison results on MVTec-AD.

	Unsupervised Methods			Self-Supervised Methods			
	[28]	[13]	[29]	[45]	[19]	[36]	Ours
carpet	77.0	99.0	**100**	95.4	80.4	82.8	99.4
grid	65.8	97.3	93.3	99.2	98.3	98.6	**99.9**
leather	92.9	**100**	**100**	95.2	**100**	88.9	**100**
tile	98.9	99.1	99.9	99.6	98.9	99.6	**100**
wood	99.2	99.0	99.5	89.3	**100**	83.2	**100**
Avg. Texture	86.8	98.9	98.5	95.7	95.5	90.6	**99.9**
bottle	98.5	**100**	**100**	99.4	99.9	93.9	99.9
cable	86.4	**95.0**	94.4	88.6	89.8	87.0	93.7
capsule	61.4	**95.2**	83.1	75.4	86.8	80.0	94.8
hazelnut	97.3	99.0	97.9	92.5	98.0	86.1	**99.8**
metal_nut	77.7	98.8	99.1	98.6	91.4	89.4	**100**
pill	65.1	94.4	90.9	92.6	90.5	86.2	**97.2**
screw	75.9	70.3	65.2	67.2	79.8	**99.8**	71.4
toothbrush	92.3	99.7	85.6	**100**	**100**	**100**	92.2
transistor	76.6	92.8	**98.0**	92.8	93.3	85.0	96.9
zipper	88.3	97.0	95.3	99.1	99.5	99.2	**99.6**
Avg. Object	82.0	94.2	91.0	90.6	92.9	90.7	**94.6**
Avg. All	87.3	95.8	93.5	92.3	93.8	90.7	**96.4**

4.2. Qualitative Results

The segmentation comparison maps in Fig. 4 illustrate that our model is capable of detecting various defect types and can accurately detect defect regions regardless of the texture classes or object classes. Although our model can not calculate diverse scores for normal and anomalous pixels, our results are closer to the ground truth compared to

Table 7. 16 shot pixel-level anomaly detection comparison results on BTAD.

	CFLOW [13]	DRAEM [45]	NSA [36]	Ours
Product 1	94.1	68.3	49.8	**96.3**
Product 2	**95.6**	77.2	92.0	93.4
Product 3	99.4	73.9	81.2	**99.4**
Average	**96.4**	73.2	74.3	**96.4**

other methods, especially on different sizes of rectangle defect regions, which are similar to our synthetic samples.

The visualization results on BTAD are shown in Fig. 5. The boundary of the defective areas is slightly blurry because we up-sampled the segmentation results to match the original mask size. However, our model can still accurately detect abnormal areas. Our anomaly segmentation map, taking the third image of Product 02 as an example, illustrates the defective areas closer to the actual anomaly than the ground truth mask.

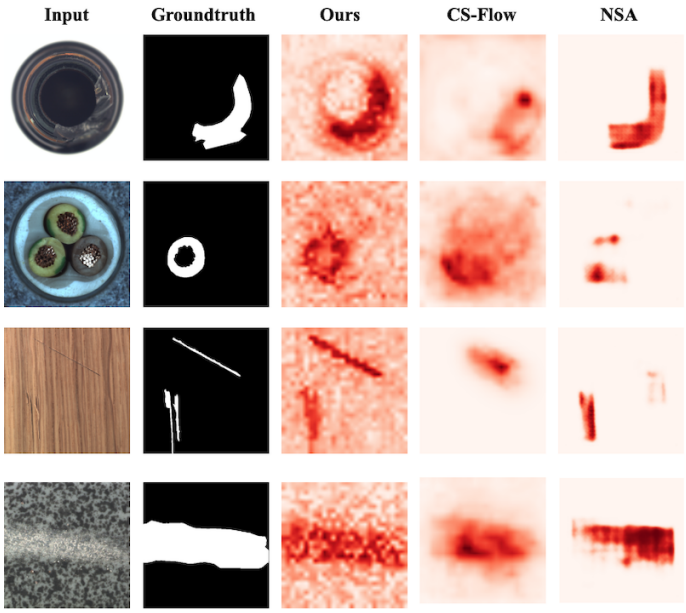

Figure 4. Pixel-level anomaly segmentation results of different classes in the MVTec-AD [4]. The bottle, cable, wood, and tile are from top to bottom. We compare our visualization results with CS-Flow [29] and NSA [36].

4.3. Complexity Analysis

We compare the model size and average inference time with other flow-based models. Note that we only calculate the parameters in flow models without the variables in the deep pretrained feature extractors and compare the total inference time, including feature extraction. Table 8 shows the comparison results. We used Intel® Core™ i7-6700 CPU @ 3.40GHz and GeForce® GTX TITAN and set the batch size to 16 in this experiment. Since we only train one flow-based model to compute the density distribution of one feature scale and the subnet in our flow model is composed of only four layers, our model size is much smaller than the others.

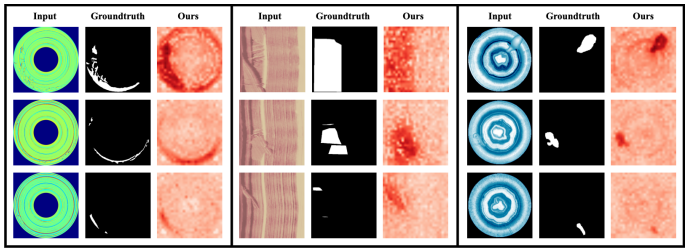

Figure 5. Examples of our anomaly segmentation results for three products of BTAD [23]. Product 01, Product 02, and Product 03 are listed from the left column to the right, respectively. We visualize the testing image, the ground truth mask, and our segmentation result.

Table 8. Flow-based model size (denoted as F-Model Size) and average inference time (denoted as Inf. Time) comparison results on class *bottle* in MVTec-AD.

	Differnet [28]	CFLOW [13]	CS-Flow [29]	Ours
N Coupling Layers	8	4	4	5
F-Model Size (M)	172.1	81.6	275.2	3.5
Inf. Time (FPS)	0.56	9.3	5.23	2.16
Inf. Time (s/img)	1.8	0.11	0.19	0.46

The results in section 4.1 show that our lightweight model can reach high performance with lesser than ten percent trainable parameters. The experimental results in section 5.1 also prove that we can improve the density estimation task on normal samples with our self-supervised loss function and additional synthesized defective samples, so our flow model can learn accurate normal distribution with the limited learning variables. However, we stack five coupling layers to build our flow-based model and use self-attention layer in our subnet, so the inference time of our lightweight model is longer than CFLOW-AD [13] and CS-Flow [29].

5. Ablation Study

5.1. Impact of Out-of-distribution Loss

Most of the existing normalizing-based models are unsupervised methods, and they are trained on defect-free im-

Table 9. Comparison results of our flow-based model with different training strategies on MVTec-AD [4]. **Bold** represent optimal results.

	Training Method	Avg. Texture	Avg. Object	Avg. All
Det	Ours (unsupervised)	99.1	96.4	97.3
	Ours (self-supervised)	**99.9**	**98.0**	**98.7**
Seg	Ours (unsupervised)	97.3	98.0	97.7
	Ours (self-supervised)	**97.8**	**98.3**	**98.1**

ages only. In this ablation experiment, we study the influence of synthetic images and our self-supervised loss. Table 9 shows the comparison results between self-supervised and unsupervised models on MVTec-AD [4].

Our self-supervised model raises 1.6% and 1.3% AUROC average score of object classes on full-dataset and 16 few-shot training scenarios. The results show that by having synthetic anomaly training images and optimizing the model on the classification task during training time, we enhance the performance of our model on both texture and object classes. Furthermore, we provide a different direction for optimizing normalizing flow-based models with the conditional self-supervised loss function. Note that our synthesized anomaly training images are produced by cutting and blending patches of normal images. Hence, the artificial defect areas are wildly different from actual anomaly types, such as *cable_swap* of *cable* and *glue* of *leather*. Moreover, our self-supervised loss function is devised to prioritize learning the distribution of normal samples, so the synthetic samples can be considered as auxiliary samples to assist learning the normal feature distribution more precisely.

5.2. Impact of Different Subnet and Residual Components

In this ablation study, we perform experiments to study the influence of different subnet. We designed three different subnet models. The first subnet model consists of three convolution layers with different kernel sizes. The second subnet model contains convolution layers with different kernel sizes and multi-head self-attention. And the last one is the architecture we use in this method, which includes additional residual components. These subnets are illustrated in Figure 6.

Table 10 summarizes the experimental results. Every flow-based model with different subnets is trained on full MVTec-AD with our self-supervised optimization mechanize. Our model with only convolution can achieve quite confirming results. However, the performance on object classes drops severely after including the multi-head self-attention layer. Although we want to increase the complexity of our subnet, the full MVTec-AD dataset provides only hundreds of training samples for each class. It does not provide enough data for training complex models like multi-head self-attention. Therefore, we join different layers with residual units to optimize the benefits of the convolution layer and the multi-head self-attention layer. The results show that our model with residual component achieves the best performance.

Table 10. Comparison results of different subnets on MVTec-AD [4]. The image-level and pixel-level results are denoted as Det and Seg, respectively.

	Subnets	Avg. Texture	Avg. Object	Avg. All
Det	Ours (convolution)	99.0	96.7	97.5
	Ours (w/o residual)	99.6	90.5	93.9
	Ours (w/ residual)	**99.9**	**98.0**	**98.7**
Seg	Ours (convolution)	97.2	98.0	97.7
	Ours (w/o residual)	97.6	97.0	97.2
	Ours (w/ residual)	**97.8**	**98.3**	**98.1**

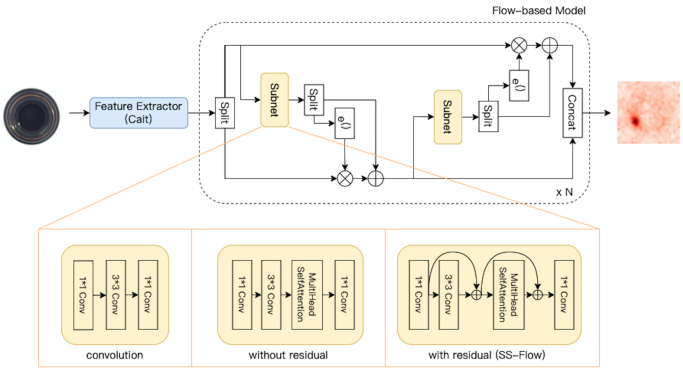

Figure 6. Ablation study of different subnets in our normalizing flow model. The above three subnet models, called as convolution, without residual, and with residual, are included in the ablation study. The results for the ablation study are given in Table 10.

6. Conclusions

In this paper, we proposed a self-supervised normalizing flow-based model that combines the advantages of the normalizing flow-based model and the self-supervised learning approach. By conditionally optimizing our model to maximize the likelihood of normal features and minimize synthetic anomaly features, we enhance the model for learning the distribution of normal features more accurately. Furthermore, we provide a different direction for optimizing normalizing flow-based models with the conditional self-supervised loss function. On the other hand, we improve the proposed model by applying a dynamic transformation subnet for our affine coupling layers. The proposed residual subnets integrate the advantages of convolution and self-attention blocks. The experimental results demonstrate that our model achieves state-of-the-art performance on several public anomaly detection benchmarks.

References

[1] 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. 1, 2

[2] Samet Akçay, Amir Atapour-Abarghouei, and Toby P Breckon. Skip-ganomaly: Skip connected and adversarially trained encoder-decoder anomaly detection. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019. 1, 2, 6

[3] Lynton Ardizzone, Till Bungert, Felix Draxler, Ullrich Köthe, Jakob Kruse, Robert Schmier, and Peter Sorrenson. Framework for easily invertible architectures (freia), 2018-2022. 5

[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. 4, 6, 7, 8

[5] 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. 1

[6] Paul Bergmann, Sindy Löwe, Michael Fauser, David Sattlegger, and Carsten Steger. Improving unsupervised defect segmentation by applying structural similarity to autoencoders. In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. SCITEPRESS - Science and Technology Publications, 2019. 1, 2

[7] Jakob Božič, Domen Tabernik, and Danijel Skočaj. End-to-end training of a two-stage neural network for defect detection. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 5619–5626. IEEE, 2021. 6

[8] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. Padim: a patch distribution modeling framework for anomaly detection and localization. In *International Conference on Pattern Recognition*, pages 475–489. Springer, 2021. 1

[9] Choubo Ding, Guansong Pang, and Chunhua Shen. Catching both gray and black swans: Open-set supervised anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7388–7398, 2022. 2

[10] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014. 3

[11] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016. 3

[12] Kota Dohi, Takashi Endo, Harsh Purohit, Ryo Tanabe, and Yohei Kawaguchi. Flow-based self-supervised density estimation for anomalous sound detection. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 336–340. IEEE, 2021. 4

[13] 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. 1, 2, 5, 6, 7

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 3

[15] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, pages 2722–2730. PMLR, 2019. 3

[16] John Taylor Jewell, Vahid Reza Khazaie, and Yalda Mohsenzadeh. One-class learned encoder-decoder network with adversarial context masking for novelty detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3591–3601, January 2022. 2

[17] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018. 3

[18] Polina Kirichenko, Pavel Izmailov, and Andrew G Wilson. Why normalizing flows fail to detect out-of-distribution data. *Advances in neural information processing systems*, 33:20578–20589, 2020. 4

[19] 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. 2, 5, 6

[20] Yufei Liang, Jiangning Zhang, Shiwei Zhao, Runze Wu, Yong Liu, and Shuwen Pan. Omni-frequency channel-selection representations for unsupervised anomaly detection. *arXiv preprint arXiv:2203.00259*, 2022. 1, 2, 6

[21] Zesheng Lin, Hongxia Ye, Bin Zhan, and Xiaofeng Huang. An efficient network for surface defect detection. *Applied Sciences*, 10(17):6085, 2020. 6

[22] Fred. A. Hamprecht. Matthias Wieler, Tobias Hahn. Weakly supervised learning for industrial optical inspection. `https://hci.iwr.uni-heidelberg.de/content/weakly-supervised-learning-industrial-optical-inspection`, 2007. 4, 6

[23] Pankaj Mishra, Riccardo Verk, Daniele Fornasier, Claudio Piciarelli, and Gian Luca Foresti. VT-ADL: A vision transformer network for image anomaly detection and localization. In *30th IEEE/IES International Symposium on Industrial Electronics (ISIE)*, June 2021. 4, 6, 7

[24] Guansong Pang, Choubo Ding, Chunhua Shen, and Anton van den Hengel. Explainable deep few-shot anomaly detection with deviation networks. *arXiv preprint arXiv:2108.00462*, 2021. 2

[25] Jonathan Pirnay and Keng Chai. Inpainting transformer for anomaly detection. In *International Conference on Image Analysis and Processing*, pages 394–406. Springer, 2022. 1, 2

[26] Masoud Pourreza, Bahram Mohammadi, Mostafa Khaki, Samir Bouindour, Hichem Snoussi, and Mohammad Sabokrou. G2d: generate to detect anomaly. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2003–2012, 2021. 2

[27] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14318–14328, June 2022. 1

[28] 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. 1, 2, 5, 6, 7

[29] 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. 1, 2, 4, 5, 6, 7

[30] Marco Rudolph, Tom Wehrbein, Bodo Rosenhahn, and Bastian Wandt. Asymmetric student-teacher networks for industrial anomaly detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2592–2602, January 2023. 1

[31] Runinho. Implementation of cutpaste. https://github.com/Runinho/pytorch-cutpaste, 2022. 5

[32] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 2

[33] Mohammadreza Salehi, Ainaz Eftekhar, Niousha Sadjadi, Mohammad Hossein Rohban, and Hamid R Rabiee. Puzzle-ae: Novelty detection in images through solving puzzles. *arXiv preprint arXiv:2008.12959*, 2020. 6

[34] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Georg Langs, and Ursula Schmidt-Erfurth. f-anogan: Fast unsupervised anomaly detection with generative adversarial networks. *Medical image analysis*, 54:30–44, 2019. 1, 2

[35] 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. 1, 2

[36] Hannah M Schlüter, Jeremy Tan, Benjamin Hou, and Bernhard Kainz. Self-supervised out-of-distribution detection and localization with natural synthetic anomalies (nsa). In *European conference on computer vision*, 2022. 2, 3, 5, 6, 7

[37] Renuka Sharma, Satvik Mashkaria, and Suyash P Awate. A semi-supervised generalized vae framework for abnormality detection using one-class classification. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 595–603, 2022. 2

[38] Jouwon Song, Kyeongbo Kong, Ye-In Park, Seong-Gyun Kim, and Suk-Ju Kang. Anoseg: Anomaly segmentation network using self-supervised learning. *arXiv preprint arXiv:2110.03396*, 2021. 2

[39] Jeremy Tan, Benjamin Hou, James Batten, Huaqi Qiu, and Bernhard Kainz. Detecting outliers with foreign patch interpolation. *arXiv preprint arXiv:2011.04197*, 2020. 2

[40] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 32–42, 2021. 5

[41] Chin-Chia Tsai, Tsung-Hsuan Wu, and Shang-Hong Lai. Multi-scale patch-based representation learning for image anomaly detection and segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3992–4000, 2022. 1

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 3

[43] Jihun Yi and Sungroh Yoon. Patch svdd: Patch-level svdd for anomaly detection and segmentation. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, November 2020. 1

[44] 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. 1, 2, 4

[45] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. Draema discriminatively trained reconstruction embedding for surface anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8330–8339, 2021. 1, 2, 5, 6, 7

[46] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. Reconstruction by inpainting for visual anomaly detection. *Pattern Recognition*, 112:107706, 2021. 1, 2