DFM: Differentiable Feature Matching for Anomaly Detection

Sheng Wu[2,*] Yimi Wang[2,*] Xudong Liu[2,*] Yuguang Yang[3,†] Runqi Wang[1,†]
Guodong Guo[4] David Doermann[5] Baochang Zhang[2,6,7,8]
[1]School of Computer Science & Technology, Beijing Jiaotong University
[2]School of Artificial Intelligence, Beihang University
[3]School of Electronic Information Engineering, Beihang University
[4]West Virginia University [5]State University of New York at Buffalo
[6]Hangzhou Research Institute, Beihang University
[7]Zhongguancun Laboratory, Beijing [8]Nanchang Institute of Technology, Nanchang

Abstract

Feature matching methods for unsupervised anomaly detection have demonstrated impressive performance. Existing methods primarily rely on self-supervised training and handcrafted matching schemes for task adaptation. However, they can only achieve an inferior feature representation for anomaly detection because the feature extraction and matching modules are separately trained. To address these issues, we propose a Differentiable Feature Matching (DFM) framework for joint optimization of the feature extractor and the matching head. DFM transforms nearest-neighbor matching into a pooling-based module and embeds it within a Feature Matching Network (FMN). This design enables end-to-end feature extraction and feature matching module training, thus providing better feature representation for anomaly detection tasks. DFM is generic and can be incorporated into existing feature-matching methods. We implement DFM with various backbones and conduct extensive experiments across various tasks and datasets, demonstrating its effectiveness. Notably, we achieve state-of-the-art results in the continual anomaly detection task with instance-AUROC improvement of up to 3.9% and pixel-AP improvement of up to 5.5%.

1. Introduction

Anomaly detection is critical for various industrial applications, where identifying deviations from normal patterns can prevent failures and ensure quality control. Feature matching-based methods like SPADE [11] and PatchCore [37] have emerged as significant advancements in this field, notably achieving excellent results on multiple key metrics on unsupervised anomaly detection settings. However, several practical limitations hinder their widespread application in some real scenes: Reliance on pre-trained backbones without adaptation for downstream tasks, especially failing to utilize available real anomalous samples. This issue also hampers the model's performance in the continual setting, where adaptation to new classes over time is essential; Inefficiency in few-shot scenarios due to the frozen backbone and oversimplified nearest neighbor matching, which necessitate the construction of a memory bank with abundant samples to maintain performance; The feature extraction and matching modules are optimized separately, resulting in incompatibility.

These shortcomings cause two main issues: feature adaptation and feature matching. Various methods have been proposed to address these problems [4, 21, 22, 49]. However, as shown in Fig. 1(a), the adaptation strategies of mainstream methods involve training the feature extractor with proxy tasks using the self-supervised framework but fail to supervise the encoder aiming for anomaly detection directly. Furthermore, the improvements in feature matching are handcrafted and static, hindering their flexibility to the extracted features as shown in Fig. 1(b). This leads to a discrepancy between feature extraction and feature matching for anomaly detection. In other words, feature matching should involve the feature learning process and, at the same time, should also be tailored to the extracted features.

To build a direct connection between feature adaptation and feature matching, we introduce the Differentiable Feature Matching (DFM) method shown in Fig. 1(c), which integrates feature extraction and feature matching in the same framework for anomaly detection and segmentation. This design allows for an end-to-end feature extractor and matching head training. Specifically, we transform the widely

∗:Co-First Authors. {shengwu, wyatt_xuanyang, tlxd}@buaa.edu.cn
†:Corresponding Authors. rqwang@bjtu.edu.cn, guangbuaa@buaa.edu.cn (Assist Corresponding)

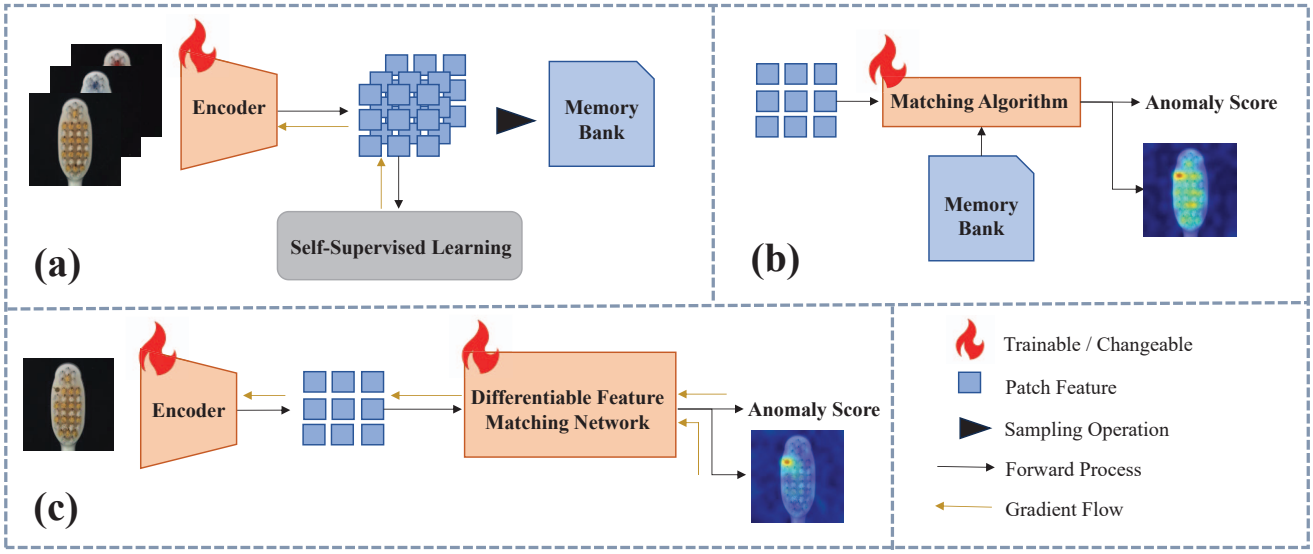

Figure 1. **Comparison of different task adaptation approaches for feature-matching-based anomaly detection models**. (a) The feature adaptation is implemented with self-supervised methods such as contrastive learning, which detaches from the ultimate goal of anomaly detection and segmentation. (b) Feature matching algorithm improvements are handcrafted and static, limiting their adaptability and performance. (c) Our proposed pipeline, where the feature extraction and matching modules are integrated into a single model, enables end-to-end optimization, thus providing better feature representation and matching schemes.

used nearest neighbor matching into a pooling-based module and embed it into a Feature Matching Network (FMN). The FMN performs a learnable transformation on the similarity matrix between the test image and the memory bank, thereby enabling more flexible feature matching. Unlike handcrafted feature matching methods, our approach emphasizes the matching learning process, thus enabling the model to capture the connections among features. As shown in Fig. 1(c), the gradient can be further propagated to the feature extractor, allowing direct supervision for anomaly detection and segmentation instead of proxy tasks.

Our framework can be incorporated into existing feature matching-based models, and we use PatchCore [37] as the base model by default. As an extension of PatchCore [37], our method preserves strong performance in unsupervised anomaly detection. Due to its learnable nature, we can utilize either real or synthetic anomalous samples for further adaptation in downstream tasks. To demonstrate the effectiveness of our approach, we conduct experiments in three settings where adaptation of the feature extractor is crucial: supervised anomaly detection, few-shot anomaly detection and continual anomaly detection. In the supervised setting, the model is trained with a small number of real anomalous samples. We employ anomaly synthesis techniques, such as CutPaste [23], to generate artificial anomalous samples for the other two settings where real anomalous samples are unavailable. Training with these augmented synthetic anomalous samples is particularly effective in a few-shot scenario, where our approach significantly outperforms the original PatchCore [37]. Our method achieves state-of-the-art performance in continual anomaly detection with an instance AUROC of 96.9% and a pixel AP of 51.1%. We attribute the excellent performance of our methods to effective feature adaptation and matching methods. The contributions of this paper are summarized as follows:

- We propose a novel differentiable feature-matching framework, DFM, which integrates the feature extraction and matching modules into a single model. This design paves a new avenue for anomaly detection model training by jointly optimizing the feature extraction and matching modules end-to-end.
- Combined with generated anomalous samples, our proposed end-to-end optimization enhances the model's capabilities across various tasks, including few-shot and continual anomaly detection.
- Our method achieves state-of-the-art performance in the continual anomaly detection task with instance AUROC improvement of up to 3.9% and pixel AP improvement of up to 5.5% over the prior art.

2. Related Work

2.1. Industrial Anomaly Detection

Anomaly detection is an important task in industrial manufacture, which aims to detect anomalous images with unusual patterns. Existing studies mainly focus on unsupervised anomaly detection, with only normal samples available during training, while both normal and anomalous

samples are included during testing.

Reconstruction-based approaches [10, 20, 52] are widely studied with advancements in generative models such as VAE [27, 28], GAN [3, 42], Normalizing Flow [35, 38] and Diffusion [48, 54]. These methods are developed based on the foundational concept that a generative model, trained on normal samples, learns to accurately reconstruct normal data while failing to reconstruct anomalies [30]. Some research also considers anomaly detection a One-Class Classification (OCC) problem and tries to find a hypersphere or superplane to distinguish abnormal images from normal ones. Representative methods include SVDD and its variants [39, 40, 51] and OCSVM [43]. Further improvements for OCC-based methods incorporate techniques such as saliency detection [5], Fourier transformation [31], and data augmentation [23]. Teacher-student architecture [6, 41, 47, 50] demonstrates strong performance in unsupervised anomaly detection tasks. In essence, a student model simulates the output of the teacher model in normal images during training. Significant discrepancies occur between the student and teacher models in anomalous images during inference, thus enabling anomaly detection.

In addition to unsupervised anomaly detection, several other related tasks have been proposed and widely studied, including few-shot anomaly detection and continual anomaly detection [26]. These settings are often closer to real-world scenarios than unsupervised settings. Few-shot anomaly detection leverages only a small number of normal samples during training and is particularly meaningful for data collection and labeling [26]. Representative works in few-shot settings include registration-based [16] and Vision-Language-Model-based methods [18]. Continual anomaly detection requires models to incrementally adapt to new anomalous classes, which mirrors real-world scenarios where manufacturing requirements evolve over time [25]. Most approaches in this area build on unsupervised anomaly detection techniques, such as reconstruction-based [45] and feature-matching-based methods [1].

2.2. Anomaly Detection via Feature Matching

Feature matching-based anomaly detection methods match features of test images to prototypes in a memory bank and evaluate the anomaly degree of test features via the Euclidean distance between test features and prototypes. SPADE [11] is an early work that stores all features of training images and uses nearest-neighbor matching to find proper prototypes of pixel features. PatchCore [37] reduces the memory bank size through coreset sampling[2] and uses patch aggregation when extracting features, marking a significant milestone in anomaly detection. Some methods optimize PatchCore by replacing the vanilla nearest neighbor matching with more efficient and effective matching algorithms. For example, CPR [24] designs a cascade matching algorithm to find more precise prototypes, and FAPM [21] constructs multiple patch-wise and layer-wise memory banks to improve inference speed and matching accuracy. Generally, most feature matching-based methods extract image features using a frozen backbone, like WideResNet [53] and ViT [14]. However, a frozen backbone may be biased and unsuitable for anomaly detection. To address this issue, some works [17, 22, 46, 51, 55] adopt self-supervised learning to extract target-oriented features. For instance, CFA [22] and Patch-SVDD [51] encourage features of normal images to be clustered within hyperspheres. At the same time, other methods [1, 17, 55] adopt a contrastive learning target for task adaptation. However, these learning targets are based on intuitively designed proxy tasks and may not align perfectly with the optimal anomaly detection and segmentation target.

3. Methodology

3.1. Feature Matching Network

We review the nearest neighbor matching algorithm commonly used in traditional feature matching-based approaches [11, 37]. A pre-trained feature extractor extracts pixel features for an input image. Then, the distance to the nearest features in a pre-built memory bank for each pixel feature is calculated as the pixel-level anomaly score. The maximum of all pixel-level anomaly scores is taken as the image-level anomaly score. This process can be mathematically expressed as follows:

$$\mathbf{S}_{i,j} = \min_{\mathbf{m}\in\mathbf{M}} ||\mathbf{P}_{i,j} - \mathbf{m}||_2, \tag{1}$$

$$s = \max_{1\le i\le H, 1\le j\le W} \mathbf{S}_{i,j}, \tag{2}$$

where $\mathbf{P}_{i,j}$ and $\mathbf{S}_{i,j}$ represent the feature and anomaly score of pixel spatially located at (i, j) respectively. s denotes the image-level anomaly score, and $\mathbf{M}$ signifies the memory bank constructed from training images. The vanilla matching process is overly simplified and will be ineffective in certain situations. For example, the same distance might indicate varying degrees of anomaly for different features. Additionally, pixels with normal appearance but abnormal position cannot be identified as anomalies. To address these issues, we propose a learnable matching module to enhance matching capability. To this end, we reformulate the nearest neighbor matching algorithm as a matrix operation and two-stage pooling:

$$\mathbf{Sim}_{i,j,k} = ||\mathbf{P}_{i,j} - \mathbf{M}_k||_2, \quad \mathbf{Sim} \in \mathbb{R}^{H\times W\times L} \tag{3}$$

$$\mathbf{S} = \text{MinPooling}(\mathbf{Sim}), \quad \mathbf{S} \in \mathbb{R}^{H\times W} \tag{4}$$

$$s = \text{MaxPooling}(\mathbf{S}), \quad s \in \mathbb{R} \tag{5}$$

Here $\mathbf{P} \in \mathbb{R}^{H\times W\times D}$ and $\mathbf{M} \in \mathbb{R}^{L\times D}$ denotes features of one test image and features in memory bank respectively. We embed the above formulation into our Feature Matching Network (FMN) as shown in Fig. 2. We also treat the

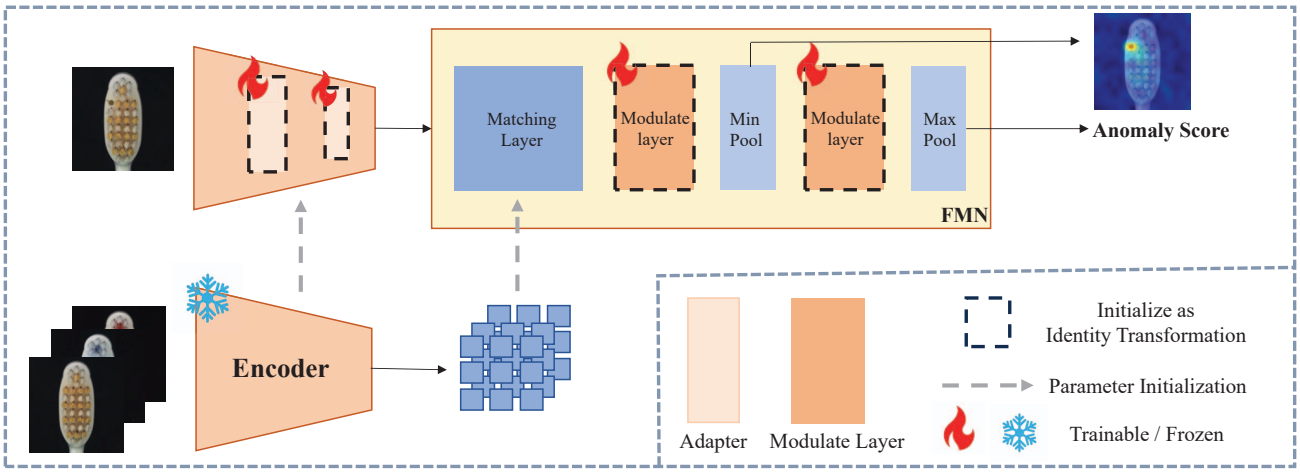

Figure 2. **Pipeline of our proposed DFM.** We replace nearest neighbor matching with a Feature Matching Network(FMN), which consists of a matching layer, two-stage pooling, and two learnable modulate layers. This design enables end-to-end optimization of the feature extractor and FMN. We insert learnable adapters into the feature extractor to facilitate efficient feature adaptation, tuning only these adapters during training. The adapters and modulate layers are initialized with identity transformation, and the matching layer is initialized with a memory bank, retaining the original capability of the base model at the start of training.

memory bank as parameters of the matching layer in FMN, which corresponds to Eq. (3). At the same time, nearest neighbor matching is replaced by two pooling layers, which correspond to Eq. (4) and Eq. (5). We insert two modulate layers before the two pooling layers to enable flexible matching. The two learnable layers can be any modules that preserve the input shape, such as linear layers, convolutional layers, or self-attention layers, and we use linear layers by default. To ensure training stability, we initialize modulate layers as identity transformation, making the FMN equivalent to the nearest neighbor matching algorithm at the start of training.

The intuition behind FMN is that the **Sim** matrix captures the similarity between the test image and the stored features, encapsulating all the information needed to determine the normality of the test image. Simple pooling operations, however, may discard valuable information. Therefore, the two learnable layers are designed to extract richer information from the **Sim** matrix. To illustrate further, consider a linear transformation as the simplest form of these layers. If the transformation matrix is diagonal, the diagonal elements can act as scaling factors, accommodating different thresholds for feature distance. Alternatively, when the transformation matrix represents a convolution operation, it can aggregate neighboring features, similar to the re-weighting trick proposed in PatchCore [37].

3.2. End-to-End Optimization of Feature Extractor

As FMN builds a differentiable connection between feature extraction and feature matching module, it allows end-to-end optimization of the feature extractor with anomaly detection and segmentation results. This approach circumvents the need for proxy self-supervised training tasks and builds a direct connection between feature adaptation and anomaly detection. To improve training efficiency, we insert adapters into the pre-trained feature extractor and fine-tune the adapters only during training. The overall model architecture is shown in Fig. 2. Similar to modulate layers in FMN, adapters in the feature extractor are initialized as identity transformation to prevent degeneration at the start of training. The training objective can be supervised or unsupervised, as discussed in the next section.

As explained in the previous section, the memory bank can be considered a parameter within the model; thus, it can also be trained. The rationale is that the features extracted from normal images with static feature extractor and sampling operation may not always be the most appropriate prototypes for anomaly recognition, particularly in restricted scenarios with few samples. The initial memory bank is constructed by extracting features from the frozen encoder, followed by coreset sampling [2] as in PatchCore [37].

Although the model is designed to be end-to-end, simultaneously optimizing the feature extractor and FMN may not yield optimal results. Continuous changes in the extracted features can result in unstable inputs to the FMN, leading to training instability. To address this issue, we employ a two-stage iterative training process. In the first stage, we optimize only the feature extractor to adapt the feature extractor for downstream tasks. In the second stage, we optimize the FMN. This approach ensures that our model starts with a baseline equivalent to PatchCore [37] but becomes more flexible and powerful with learnable feature extraction and feature matching modules.

如何先训练的特征提取器

3.3. Training Paradigm

Training with supervised target. The learnable nature of our method enables the utilization of available anomaly samples in real-world scenarios, which cannot be used in traditional feature matching-based methods [11, 37]. Additionally, we can generate artificial anomalous samples using techniques like CutPaste [23]. Suppose we have anomaly dataset $I_a = \{I_a^i | 1 \leq i \leq N_a\}$ and normal dataset $I_n = \{I_n^i | 1 \leq i \leq N_n\}$, we use targets coming from SVDD loss [40] and deviation loss [32] for training:

$$\mathcal{L}_{\text{detec}} = \frac{1}{N_a} \sum_{i=1}^{N_a} \max(0, s^*_{\text{detec}} - s_a^i) + \frac{1}{N_n} \sum_{i=1}^{N_n} ||s_n^i - \mu_{norm}||_1, \quad (6)$$

where s^i denotes the anomaly score of i-th image and s^*_{detec} represent reference score, encouraging the model to focus on hard samples *i.e.* anomaly samples with a low score. μ_{norm} is the mean score value of normal samples used to cluster normal scores. For simplicity, the reference score is defined as $s^*_{\text{detec}} = \beta \cdot \max(s_a^i)$ where β is a scaling factor set between ~~0 and 1, with a de~~fault value of 0.9.

When segmentation masks for anomaly images $\mathbf{M}_a = \{\mathbf{M}_a^i \in \mathbb{R}^{H \times W} | 1 \leq i \leq N_a\}$ are available, we can incorporate a segmentation loss, similar to the detection loss, into the training objective, which is formulated as follows:

$$\mathcal{L}_{\text{seg}} = \frac{1}{N_a \cdot HW} \sum_{i=1}^{N_a} \mathbf{M}_a^i \cdot \max(0, s^*_{seg} - \mathbf{S}_a^i) \quad (7)$$

where s^*_{seg} and $\mathbf{S}_a^i \in \mathbb{R}^{H \times W}$ denotes the reference score anomaly score for segmentation respectively, and we set it as β times the maximum pixel-level anomaly score. The complete supervised training target is expressed as follows:

$$\mathcal{L}_{\text{sup}} = \lambda_{\text{detec}} \cdot \mathcal{L}_{\text{detec}} + \lambda_{\text{seg}} \cdot \mathcal{L}_{\text{seg}}. \quad (8)$$

Training with unsupervised target. In the unsupervised target, we use only normal samples for training, an effective practice. Compared to existing self-supervised methods that exclude anomalous samples, our loss function is calculated directly from detection and segmentation results, thereby providing more accurate supervisory signals. The unsupervised training target is formulated as follows:

$$\mathcal{L}_{\text{unsup}} = \frac{1}{N_n} \sum_{i=1}^{N_n} \max(0, s_n^i - s^*_{\text{detec}}) + \frac{1}{N_n \cdot H \cdot W} \sum_{i=1}^{N_n} \max(0, \mathbf{S}_n^i - s^*_{\text{seg}}) \quad (9)$$

In our two-stage interactively training, we use an unsupervised learning objective in the first stage for preliminary adaptation and a supervised learning objective with real or artificially generated anomaly samples in the second stage for further adaptation.

4. Experiment

4.1. Implementation Details

Datasets. Our experiments utilize **MVTec AD** [7] and **VisA** [55] datasets. The MVTec AD dataset comprises 15 different categories with 3629 images in the training set and 1725 images in the testing set. Only normal images are available in the training set, while the testing set includes both normal and anomalous images. VisA dataset consists of 12 different object types and 10821 images, including 9621 normal samples and 1200 anomalous samples. For supervised setting, we adhere to the official split of the VisA dataset, using a total of 5773 normal images and 720 anomalous images for training. We use the VisA dataset in the supervised setting, while other experiments are conducted on the MVTec AD dataset. Results for continual and few-shot settings on VisA and additional datasets are provided in the appendix.

Metrics and Baselines. For anomaly detection, we report the *Instance-level Area Under the Receiver Operating Characteristic* (I-AUROC) in accordance with the literature [11, 37]. For anomaly segmentation, we report the *pixel-wise AUROC* (P-AUROC) and *pixel-wise Average Precision* (P-AP), similar to the evaluation of anomaly detection. For continual anomaly detection, we additionally employ the *Forgetting Measure* (FM) to assess the model's performance, following the methodology outlined in previous works [1, 25]. We mainly compare our method against feature-matching-based methods [11, 22, 37, 49]. Besides, we incorporate comparison with other representative methods in specific settings like WinCLIP [18] in few-shot anomaly detection and DNE [25] in continual anomaly detection. Given that our approach is built upon PatchCore [37], a direct comparison with it is particularly significant.

Experiment settings. In all experiments, we use ViT-B/16 from CLIP [34] as the feature extractor, and the output of the 6th block as extracted patch features following [1] unless otherwise specified. Additional experiments utilizing different feature extractors are presented in the ablation study. In default, we employ linear layers for the adapter and modulate layers in FMN. The default size of the memory bank is set to 196×768, following [1]. All experiments are conducted on a single NVIDIA RTX-3090.

4.2. Results of Continual Anomaly Detection

Continual anomaly detection [1, 25, 45] requires the model to learn to detect anomalies of different object categories incrementally. The challenge lies in effectively integrating

Classes	CFA [22]	PaDiM [12]	SPADE [11]	DNE [25]	PatchCore [37]	UCAD [1]	DFM
Bottle	30.9, 6.80	45.8, 7.20	30.2, 12.2	99.0, -	53.3, 8.70	**100**, 75.2	99.7, **76.8**
cable	48.9, 5.60	54.4, 3.70	44.4, 5.20	61.9, -	50.5, 4.30	75.1, 29.0	**94.8**, **50.6**
capsule	27.5, 5.00	41.8, 3.00	52.5, 4.40	60.9, -	35.1, 4.20	86.6, **34.9**	**99.6**, 24.1
carpet	83.4, 27.1	45.4, 2.30	52.9, 11.7	98.4, -	86.5, 40.7	96.5, 62.2	**99.9**, **77.1**
grid	57.1, 0.40	70.4, 0.60	46.0, 0.40	**99.8**, -	72.3, 0.30	94.4, 18.7	99.0, **22.8**
hazelnut	90.3, 34.1	63.5, 18.3	41.0, 51.2	92.4, -	95.9, 44.3	**99.4**, **50.6**	97.7, 47.9
leather	93.5, 39.3	41.8, 3.90	57.7, 26.4	**100**, -	85.4, 35.2	**100**, 33.3	**100**, **43.2**
metal_nut	46.4, 25.5	44.6, 15.5	59.2, 18.1	98.9, -	45.6, 18.9	98.8, **77.5**	**100**, 69.0
pill	52.8, 8.00	44.9, 4.40	48.4, 6.00	67.1, -	51.1, 5.80	89.4, **63.4**	**98.3**, 57.6
screw	52.8, 1.50	57.8, 1.40	51.4, 2.00	58.8, -	62.6, 1.70	73.9, 21.4	**76.5**, **24.2**
tile	76.3, 15.5	58.1, 9.60	88.1, 9.60	98.0, -	74.8, 12.4	**99.8**, 54.9	98.2, **62.3**
toothbrush	51.9, 5.30	67.8, 4.40	38.6, 4.30	93.3, -	60.0, 2.80	**100**, 29.8	99.7, **33.1**
transistor	32.0, 5.60	40.7, 4.90	62.2, 5.00	87.7, -	42.7, 5.30	87.4, 39.8	**93.2**, **50.1**
wood	92.3, 28.1	54.9, 8.00	89.7, 17.2	93.0, -	90.0, 27.0	**99.5**, 53.5	98.6, **58.1**
zipper	98.4, 57.3	85.5, 45.2	94.9, 53.1	95.8, -	97.4, **60.4**	93.8, 39.8	**98.7**, 51.1
average	62.3, 17.7	54.5, 8.60	57.1, 15.1	87.0, -	66.9, 18.1	93.0, 45.6	**96.9**, **51.1**
avg FM	36.1, 8.30	36.8, 36.6	28.5, 31.9	11.6, -	31.8, 34.3	**1.00**, **1.30**	1.50, **1.30**

Table 1. Instance AUROC↑ (left), Pixel AP↑ (right), and corresponding FM↓ on MVTec AD dataset after training on the last subdataset. The best results are highlighted in bold. Our method achieves SOTA accuracy in both Instance AUROC and Pixel AP.

continual learning with anomaly detection. To apply DFM to continual scenarios, we employ a similar framework in UCAD [1]. To be more specific, we train adapters for each class and extract features of each class using a frozen backbone to form class-specific features. During inference, we distinguish between different classes by extracting features of the test image with a frozen encoder and retrieving the adapter corresponding to the nearest class-specific features. Although UCAD [1] also proposes to adapt frozen backbone in a continual setting, it utilizes contrastive learning with proxy tasks instead of direct supervision from anomaly detection and segmentation, resulting in inferior feature representation. The results presented in Tab. 1 indicate that our model achieves state-of-the-art accuracy in I-AUROC and P-AP, demonstrating the effectiveness of our framework for adapting feature extractors in continual anomaly detection.

4.3. Results of Few-Shot Anomaly Detection

In the few-shot setting, the training dataset comprised a limited set of normal samples, specifically 1, 2, 4, or 8 instances in our experiments. As shown in Tab. 2, We achieve clear improvement compared to the original PatchCore [37] in all shots. We attribute the inferior performance of PatchCore to its frozen backbone and static matching algorithm, which are ineffective in extracting appropriate features when training samples are limited, as discussed in GraphCore [49]. In contrast, our method provides an effective way to adapt a frozen backbone to restricted downstream tasks. Across all shots, our method demonstrates performance equivalent to or surpassing methods specially designed for few-shot scenarios like GraphCore and WinCLIP [18] in anomaly detection and segmentation, demonstrating the effectiveness of our approach.

4.4. Results of Supervised Anomaly Detection

Existing feature matching-based methods cannot utilize a small number of anomalous samples present in real-world scenarios, which we have proven to be very useful. As shown in Fig. 3, a small number of real anomalous samples can greatly improve the model's performance, achieving approximately 4% higher accuracy than the original PatchCore in both I-AUROC and P-AP. Additionally, we compare the effectiveness of real anomalous samples and artificial anomalous samples generated through CutPaste [23]. As shown in Fig. 3, real anomalous samples are significantly more beneficial, especially in the later epochs, where the performance gap between the model trained with real anomalous samples and the model trained with artificially generated anomalous samples keeps widening.

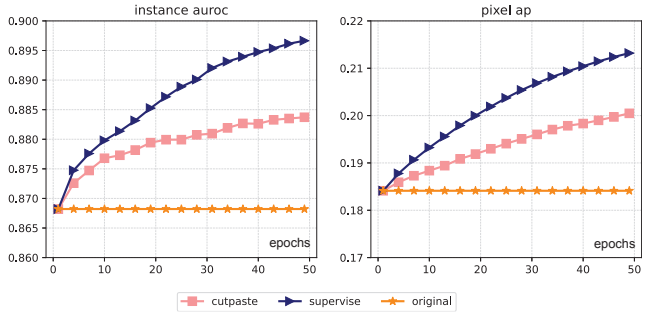

Figure 3. Instance AUROC↑ and Pixel AP↑ on supervised setting using VisA dataset. The term 'supervise' in the figure indicates utilizing real anomalous samples for training.

Instance-AUROC							
K	GraphCore [49]	CFA [22]	WinCLIP [18]	PatchCore [37]	RegAD [16]	DFM-CFA	DFM-PatchCore
1	89.9	78.8	**93.1**	78.5	82.4	87.2	93.0
2	91.9	81.1	94.4	87.8	85.7	90.2	**95.0**
4	92.9	85.0	95.2	89.5	88.2	92.0	**95.9**
8	95.9	90.9	-	94.3	91.2	93.5	**96.2**
Pixel-AUROC							
K	GraphCore [49]	CFA [22]	WinCLIP [18]	PatchCore [37]	RegAD [16]	DFM-CFA	DFM-PatchCore
1	95.6	90.7	95.2	90.1	-	95.2	**96.3**
2	96.9	91.7	96.0	94.8	94.6	96.0	**97.0**
4	**97.4**	91.3	96.2	95.0	95.8	96.2	96.7
8	**97.8**	91.6	-	95.6	96.7	96.2	96.3

Table 2. Comparison of Instance AUROC↑ and pixel AUROC↑ on few-shot settings using MVTec AD dataset.

1,2	3,4	5,6	1shot I-AUROC	1shot P-AUROC	2shot I-AUROC	2shot P-AUROC
✓	✓	✓	86.5	95.8	94.6	96.9
✓			86.7	96.0	92.7	96.2
	✓		91.0	**96.4**	94.4	96.6
		✓	**93.0**	96.3	**95.0**	**97.0**

Table 3. Ablation of different adapter layers on few-shot settings using MVTec AD dataset.

4.5. Analysis and Ablation Study

Ablation of different base models. PatchCore is used as the default base model in all experiments due to its simplicity and robust performance. We further conduct experiments on few-shot and continual settings with CFA [22], which is another effective feature-matching-based method. As shown in Tab. 2 and Tab. 4, our method can significantly enhance the performance of CFA in few-shot setting and achieve high performance in continual settings, demonstrating the generalization of our framework.

Method	Instance-AUROC	Pixel-AP
UCAD	93.0	45.6
DFM-CFA	96.2	48.0
DFM-PatchCore	96.9	51.0

Table 4. Comparison between previous SOTA and our method with different base models in continual setting on MVTec AD.

Ablation of learnable structures. To determine the importance of each part in our framework, we evaluate anomaly detection performance with different learnable structures and learning strategies on the few-shot setting using the MVTec AD [7] dataset. Adapters are inserted into 1st and 2nd layers here. As shown in Tab. 5, the FMN and adapters can separately improve the performance of the original PatchCore, 3% for FMN and 6% for adapters in I-AUROC. As mentioned earlier, direct joint optimization of FMN and adapters could result in unstable training, thus leading to inferior performance. However, the performance can be improved further with iterative training of FMN and adapters, proving the effectiveness of aligning the feature extraction and feature matching module.

FMN	Adapter	Iterative	I-AUROC	P-AUROC
frozen backbone			85.43	95.87
✓			88.39	95.02
	✓		91.73	96.17
✓	✓		84.27	90.23
✓	✓	✓	**92.73**	**96.20**

Table 5. Ablation of different learnable structures and training strategies on 2-shot setting using MVTec AD dataset.

Ablation of different backbones. We implement our framework in different backbones, including larger ViT [14] and WideResNet [53]. The experiment is conducted on a two-shot setting using the MVTec AD dataset. We finetune FMN only here. As shown in Tab.6, the performance of different backbones all improve significantly, indicating the efficiency of our method for feature matching based method in few-shot anomaly detection.

Ablation of adapter layers. The transformer blocks of ViT-B are divided into three segments to represent varying degrees of layer depth. The experiment is carried out under a few-shot setting. As shown in Tab. 3, the block where the adapters are inserted can greatly affect the performance. More specifically, the performance is better if the adapters are inserted into deeper layers of the model.

Visualization of segmentation results. We visualize anomaly segmentation results on the MVTec AD dataset.

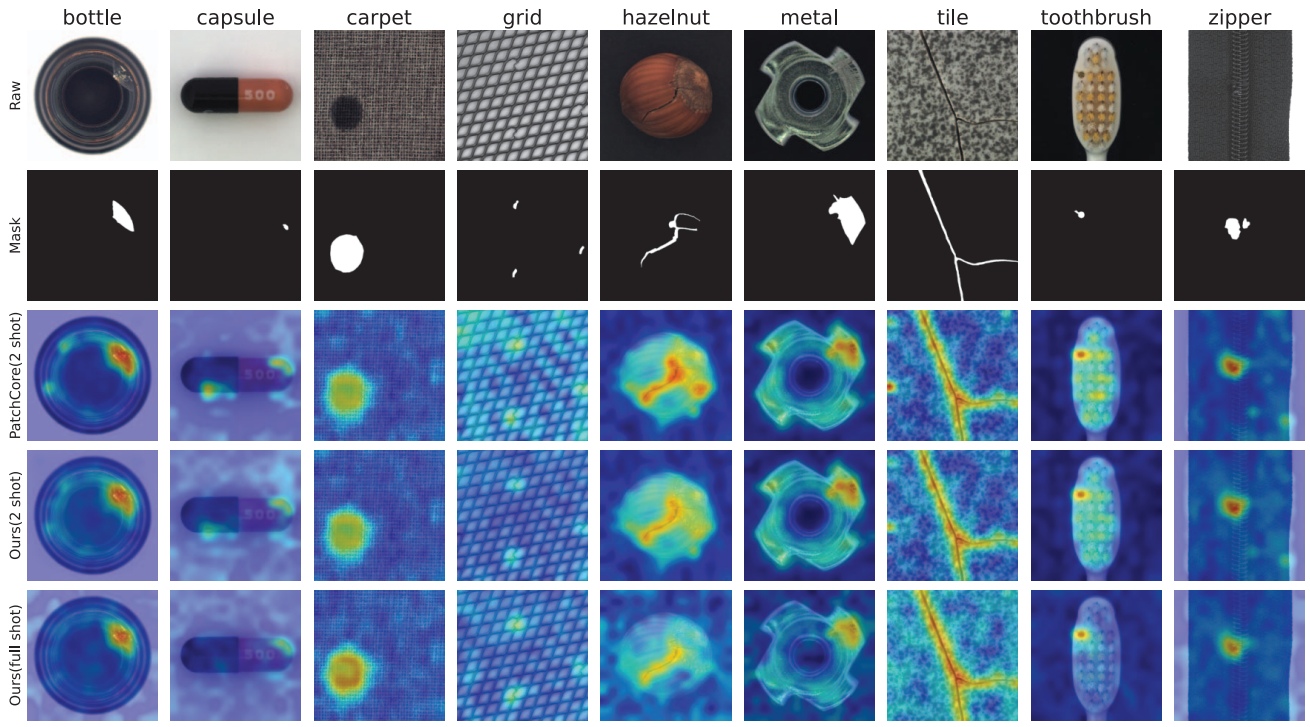

Figure 4. Anomaly segmentation comparison between our method and PatchCore [37].

Backbone	Method	I-AUROC	P-AP
ViT-B/16	original	85.4	42.7
	trained	88.4(+3.0)	43.8(+1.1)
ViT-L/14	original	81.4	41.7
	trained	84.9(+3.5)	43.5(+1.8)
WideResNet50-2	original	85.4	44.2
	trained	87.7(+2.3)	44.8(+0.6)

Table 6. Ablation of backbones in a 2-shot setting on MVTec AD.

Both PatchCore [37] and our method utilize ViT-L [14] as the backbone. As shown in Fig.4, our method demonstrates a strong capability for accurate localization and generates segmentation maps with less noise compared to PatchCore, consistent with quantitative results.

Method	FPS Train / Infer	Memory Consumption (MB) Train / Infer
PatchCore	- , 20.32	- , 2713
DFM	24.09 , 49.90	4923 , 3671

Table 7. FPS↑ and Memory Consumption↓ comparison between PatchCore and our method during training and inference.

Training and inference efficiency analysis As shown in Tab.7, our method has a clear advantage in inference time, albeit at higher GPU memory consumption. The critical difference lies in the matching process: our approach utilizes FMN, which is based on matrix operations, while PatchCore relies on a vector search algorithm implemented using the Faiss [19]. Since modern graphic cards support parallel computing for matrix operations, it's unsurprising that our method demonstrates superior running efficiency. Although our method requires more memory during training and inference, the memory cost is acceptable in improving overall performance and inference efficiency.

5. Conclusion

In this paper, we develop a feature matching-based anomaly detection method, which can end-to-end train feature extraction and feature matching modules. We achieve a Differentiable Feature Matching (DFM) method to integrate the feature extraction and matching modules in the same framework. The core component of our framework is the Feature Matching Network (FMN), which changes feature matching into a differentiable module, enabling joint optimization of the feature extractor and feature matching module. To validate the effectiveness of our method, we implement DFM in different base models. The results show that our framework can improve the performance of the feature matching-based methods in various tasks, specifically in few-shot anomaly detection and continual anomaly detection.

6. Acknowledgements

The work was supported by the National Key Research and Development Program of China (Grant No. 2023YFC3306401), and Zhejiang Provincial Natural Science Foundation of China under Grant No. LD24F020007, Beijing Natural Science Foundation L223024, L244043 and Z241100001324017 "One Thousand Plan" projects in Jiangxi Province Jxsq2023102268 and the Beijing Jiaotong University "Jingying Plan" No.K24XKRC00130.

References

[1] Unsupervised continual anomaly detection with contrastively-learned prompt. In *AAAI*, 2024. 3, 5, 6, 11

[2] Pankaj K Agarwal, Sariel Har-Peled, Kasturi R Varadarajan, et al. Geometric approximation via coresets. *Combinatorial and computational geometry*, 52(1):1–30, 2005. 3, 4

[3] Samet Akçay, Amir Atapour-Abarghouei, and T. Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *ACCV*, 2018. 3

[4] Jaehyeok Bae, Jae-Han Lee, and Seyun Kim. Pni: industrial anomaly detection using position and neighborhood information. In *ICCV*, 2023. 1

[5] Xiaolong Bai, Yuming Fang, Weisi Lin, Lipo Wang, and Bing-Feng Ju. Saliency-based defect detection in industrial images by using phase spectrum. *IEEE Transactions on Industrial Informatics*, 10(4):2135–2145, 2014. 3

[6] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings. *CVPR*, 2019. 3

[7] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection. In *CVPR*, 2019. 5, 7, 11, 12

[8] Paul Bergmann, Kilian Batzner, Michael Fauser, David Sattlegger, and Carsten Steger. Beyond dents and scratches: Logical constraints in unsupervised anomaly detection and localization. *International Journal of Computer Vision*, 130 (4):947–969, 2022. 11, 12

[9] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. 11

[10] J.K. Chow, Z. Su, J. Wu, P.S. Tan, X. Mao, and Y.H. Wang. Anomaly detection of defects on concrete structures with the convolutional autoencoder. *Advanced Engineering Informatics*, 45:101105, 2020. 3

[11] Niv Cohen and Yedid Hoshen. Sub-image anomaly detection with deep pyramid correspondences. *arXiv preprint arXiv:2005.02357*, 2020. 1, 3, 5, 6

[12] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. Padim: a patch distribution modeling framework for anomaly detection and localization. In *ICPR*, 2021. 6

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 11

[14] 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. *arXiv preprint arXiv:2010.11929*, 2020. 3, 7, 8, 11

[15] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 11

[16] Chaoqin Huang, Haoyan Guan, Aofan Jiang, Ya Zhang, Michael Spratling, and Yan-Feng Wang. Registration based few-shot anomaly detection. In *ECCV*, 2022. 3, 7

[17] Jeeho Hyun, Sangyun Kim, Giyoung Jeon, Seung Hwan Kim, Kyunghoon Bae, and Byung Jun Kang. Reconpatch: Contrastive patch representation learning for industrial anomaly detection. In *WACV*, 2024. 3

[18] Jongheon Jeong, Yang Zou, Taewan Kim, Dongqing Zhang, Avinash Ravichandran, and Onkar Dabeer. Winclip: Zero-/few-shot anomaly classification and segmentation. In *CVPR*, 2023. 3, 5, 6, 7, 11

[19] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019. 8

[20] Gaoqiang Kang, Shibin Gao, Long Yu, and Dongkai Zhang. Deep architecture for high-speed railway insulator surface defect detection: Denoising autoencoder with multitask learning. *IEEE Transactions on Instrumentation and Measurement*, 68(8):2679–2690, 2019. 3

[21] Donghyeong Kim, Chaewon Park, Suhwan Cho, and Sangyoun Lee. Fapm: Fast adaptive patch memory for real-time industrial anomaly detection. In *ICASSP*, 2023. 1, 3

[22] Sungwook Lee, Seunghyun Lee, and Byung Cheol Song. Cfa: Coupled-hypersphere-based feature adaptation for target-oriented anomaly localization. *IEEE Access*, 10: 78446–78454, 2022. 1, 3, 5, 6, 7

[23] Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. In *CVPR*, 2021. 2, 3, 5, 6

[24] Hanxi Li, Jianfei Hu, Bo Li, Hao Chen, Yongbin Zheng, and Chunhua Shen. Target before shooting: Accurate anomaly detection and localization under one millisecond via cascade patch retrieval. *arXiv preprint arXiv:2308.06748*, 2023. 3

[25] Wujin Li, Jiawei Zhan, Jinbao Wang, Bizhong Xia, Bin-Bin Gao, Jun Liu, Chengjie Wang, and Feng Zheng. Towards continual adaptation in industrial anomaly detection. In *ACM MM*, 2022. 3, 5, 6

[26] Jiaqi Liu, Guoyang Xie, Jinbao Wang, Shangnian Li, Chengjie Wang, Feng Zheng, and Yaochu Jin. Deep industrial image anomaly detection: A survey. *Machine Intelligence Research*, 21(1):104–135, 2024. 3

[27] Wenqian Liu, Runze Li, Meng Zheng, Srikrishna Karanam, Ziyan Wu, Bir Bhanu, Richard J. Radke, and Octavia I. Camps. Towards visually explaining variational autoencoders. *CVPR*, 2019. 3

[28] Takashi Matsubara, Kazuki Sato, Kenta Hama, Ryosuke Tachibana, and Kuniaki Uehara. Deep generative model using unregularized score for anomaly detection with heterogeneous complexity. *IEEE Transactions on Cybernetics*, 52 (6):5161–5173, 2022. 3

[29] 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 *ISIE*, 2021. 11, 12

[30] Arian Mousakhan, Thomas Brox, and Jawad Tayyub. Anomaly detection with conditioned denoising diffusion models. *arXiv preprint arXiv:2305.15956*, 2023. 3

[31] Menghui Niu, Kechen Song, Liming Huang, Qi Wang, Yunhui Yan, and Qinggang Meng. Unsupervised saliency detection of rail surface defects using stereoscopic images. *IEEE Transactions on Industrial Informatics*, 17(3):2271–2281, 2021. 3

[32] Guansong Pang, Chunhua Shen, and Anton Van Den Hengel. Deep anomaly detection with deviation networks. In *ACM SIGKDD*, 2019. 5

[33] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 2019. 11

[34] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 5, 11

[35] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *ArXiv*, abs/1505.05770, 2015. 3

[36] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951. 11

[37] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. In *CVPR*, 2022. 1, 2, 3, 4, 5, 6, 7, 8, 11, 12, 13

[38] Marco Rudolph, Bastian Wandt, and Bodo Rosenhahn. Same same but differnet: Semi-supervised defect detection with normalizing flows. In *WACV*, 2021. 3

[39] Lukas Ruff, Nico Görnitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Robert A. Vandermeulen, Alexander Binder, Emmanuel Müller, and M. Kloft. Deep one-class classification. In *ICML*, 2018. 3

[40] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *ICML*, 2018. 3, 5

[41] Mohammadreza Salehi, Niousha Sadjadi, Soroosh Baselizadeh, Mohammad H. Rohban, and Hamid R. Rabiee. Multiresolution knowledge distillation for anomaly detection. In *CVPR*, 2021. 3

[42] Thomas Schlegl, Philipp Seeböck, Sebastian M. Waldstein, Ursula Margarethe Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *Information Processing in Medical Imaging*, 2017. 3

[43] Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001. 3

[44] shirowalker. Ucad. https://github.com/shirowalker/UCAD, 2023. Accessed: 2025-03-10. 11

[45] Jiaqi Tang, Hao Lu, Xiaogang Xu, Ruizheng Wu, Sixing Hu, Tong Zhang, Tsz Wa Cheng, Ming Ge, Ying-Cong Chen, and Fugee Tsung. An incremental unified framework for small defect inspection. *arXiv preprint arXiv:2312.08917*, 2023. 3, 5

[46] Chin-Chia Tsai, Tsung-Hsuan Wu, and Shang-Hong Lai. Multi-scale patch-based representation learning for image anomaly detection and segmentation. In *WACV*, 2022. 3

[47] Guodong Wang, Shumin Han, Errui Ding, and Di Huang. Student-teacher feature pyramid matching for anomaly detection. In *BMVC*, 2021. 3

[48] Julian Wyatt, Adam Leach, Sebastian M Schmon, and Chris G Willcocks. Anoddpm: Anomaly detection with denoising diffusion probabilistic models using simplex noise. In *CVPR*, 2022. 3

[49] Guoyang Xie, Jinbao Wang, Jiaqi Liu, Feng Zheng, and Yaochu Jin. Pushing the limits of fewshot anomaly detection in industry vision: Graphcore. *arXiv preprint arXiv:2301.12082*, 2023. 1, 5, 6, 7

[50] Shinji Yamada and Kazuhiro Hotta. Reconstruction student with attention for student-teacher pyramid matching. *ArXiv*, abs/2111.15376, 2021. 3

[51] Jihun Yi and Sungroh Yoon. Patch svdd: Patch-level svdd for anomaly detection and segmentation. In *ACCV*, 2018. 3

[52] Sanyapong Youkachen, Miti Ruchanurucks, Teera Phatrapomnant, and Hirohiko Kaneko. Defect segmentation of hot-rolled steel strip surface by using convolutional auto-encoder and conventional image processing. In *IC-ICTES*, 2019. 3

[53] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 3, 7, 11

[54] Xinyi Zhang, Naiqi Li, Jiawei Li, Tao Dai, Yong Jiang, and Shu-Tao Xia. Unsupervised surface anomaly detection with diffusion probabilistic model. In *ICCV*, 2023. 3

[55] Yang Zou, Jongheon Jeong, Latha Pemula, Dongqing Zhang, and Onkar Dabeer. Spot-the-difference self-supervised pre-training for anomaly detection and segmentation. In *ECCV*, 2022. 3, 5, 11, 12

[56] zqhang. Accurate-winclip-pytorch. https://github.com/zqhang/Accurate-WinCLIP-pytorch, 2023. Accessed: 2025-03-10. 11