Unsupervised Image Anomaly Detection and Localization in Industry Based on Self-Updated Memory and Center Clustering

Yongheng Liu, Xiangdong Gao, James Zhiqing Wen, and Huiyuan Luo

***Abstract*— Defect detection of industrial products often uses computer vision methods. Detecting anomalies in the image can reflect the defect of the product. To adapt to the scene of less defect samples and unclear defect classification standards in industrial production and improve the accuracy and robustness of detection, this article proposes a new unsupervised anomaly detection and localization framework based on self-updated memory and center clustering (SMCC). Distinct from previous works, it uses a pretrained model to extract image features, and then uses a Gaussian mixture model to cluster and obtain cluster centers, so that normal sample features are compactly distributed around the cluster centers, thereby better distinguishing normal and abnormal sample features. The advantage of the self-updated memory bank is to reduce the use of memory and adjust the parameters of the pretrained network to make it more suitable for the distribution of the current dataset. Our experiments on the MVTec AD and other datasets show the effectiveness of SMCC for anomaly detection and localization.**

***Index Terms*— Anomaly detection, anomaly localization, Gaussian mixture model.**

I. Introduction

DEFECT detection [1] is one of the important technologies to ensure product quality, which aims to find the appearance defects of various industrial products. Although the manufacturing process can be improved by signal processing and filtering to remove noise [2], [3], [4], [5], [6], the products still suffer from various defects. The quality of the product is usually judged by visual imaging and further processing of the image [7], [8], [9] when detecting product surface defects. Recently, deep learning is gradually applied to industrial detection tasks [10], [11] because of the shortcomings of traditional visual detection, such as low accuracy and poor adaptability. Detection algorithms based on deep learning require a large number of defect samples for training. However, in actual industrial production, the number of defect samples is few and uneven, and it is often difficult to establish a clear classification criterion. The previously unknown and missed defects can make the classification task even more difficult.

Anomaly detection is to identify and separate data that are significantly different from normal data. The anomaly of industrial products refers to any defect that occurs outside the normal product. This article focuses on the surface defects of industrial products and does not discuss internal defects. The purpose of anomaly detection is to determine whether the sample is abnormal, and the purpose of anomaly location is to find the location of the anomaly. Anomaly detection and location of industrial products can be achieved by acquiring its surface image and detecting the image with an algorithm. Currently, the unsupervised anomaly detection algorithm is widely studied in industry because its training is only for normal samples without defect samples [12]. The purpose of these algorithms is to accurately detect and locate anomaly areas in the image using the prior knowledge of anomaly-free images. This task is of particular importance in the intelligent manufacturing process of qualified products, such as automatic inspection and screening of defects or defective products [13]. The unsupervised anomaly detection algorithm not only resolves the problem of small samples in industrial scenes but also does not need expensive sample labeling for negative samples.

At present, the main challenges of unsupervised anomaly detection are as follows.

1) There are few datasets available in this field.
2) When there is noise in normal samples, the accuracy of the anomaly detection algorithm trained by normal samples will be greatly reduced.
3) Anomaly detection for complex background and target is difficult.

Previous work on anomaly detection is mostly based on reconstruction and representation. The method based on reconstruction aims to establish the comparison between the samples before and after reconstruction [14], [15]. Its idea is to use only normal data to train the reconstruction network, while the input anomaly image cannot be reconstructed well, and the difference can be transformed into the anomaly score for a detection task. The reconstruction function is usually realized by autoencoders and teacher–student networks [16]. However, the disadvantage of such methods is that the reconstruction

Manuscript received 18 January 2023; revised 29 March 2023; accepted 13 April 2023. Date of publication 2 May 2023; date of current version 11 May 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 52275317, in part by the Guangdong Provincial Natural Science Foundation of China under Grant 2023A1515012172, in part by the Guangzhou Municipal Special Fund Project for Scientific and Technological Innovation and Development under Grant 2023B03J1326, in part by the Special Project for Tackling Key Scientific and Technological Problems in the Industrial Field of Foshan in 2020 under Grant 2020001006297, and in part by the Shunde District's 2020 Core Technology Research Projects under Grant 2030218000174. The Associate Editor coordinating the review process was Dr. Hongrui Wang. *(Corresponding author: Xiangdong Gao.)*

Yongheng Liu and Xiangdong Gao are with the Guangdong Provincial Welding Engineering Technology Research Center, Guangdong University of Technology, Guangzhou 510006, China (e-mail: liuyh_gdut@163.com; gaoxd@gdut.edu.cn).

James Zhiqing Wen and Huiyuan Luo are with the Ji Hua Laboratory (Advanced Manufacturing Science and Technology GuangDong Laboratory), Foshan 528200, China.

Digital Object Identifier 10.1109/TIM.2023.3271754

of normal images is often fuzzy, and normal samples may be reconstructed as anomaly or the opposite situation may occur when the reconstruction ability is strong. Recently, the normalizing flow (NF) [17] has been increasingly used in the field of image anomaly detection [18], [19]. The NF model transforms arbitrary complex data distribution into some basic simple distribution (such as single Gaussian distribution and uniform distribution) by constructing a reversible transformation function. The disadvantage is that it is often accompanied by a large amount of memory consumption and network computation, and it requires more training time.

Based on this background, this article proposes a novel framework based on self-updated memory and center clustering (SMCC) for unsupervised anomaly detection and location.

II. Related Work

Although there are many literatures related to learning normal representation or reconstruction, some new studies have focused on pretrained models in recent years due to some shortcomings mentioned above and the scarcity of industrial anomaly detection datasets. These works use pretrained network models on large external natural image datasets (such as ImageNet [20]) as feature extractors so that the data at hand do not require additional adaptation. On this basis, a series of anomaly detection methods are generated, which rely on better reuse of the features extracted by the pretrained model.

For example, some methods apply classification to the extracted features. The deep one-class classification (OCC) method first learns the data description of the normal sample, and then uses a criterion (such as the distance to the class center) to detect and locate anomalies in the test sample. Although the principle is similar, unlike the traditional support vector machine (SVM) method [21], the support vector domain description (SVDD) [22] can be used as an unsupervised method to solve OCC problems. Some methods are based on SVDD and combined with deep neural networks, such as DeepSVDD [23] and patchSVDD [24]. Some methods store the features of the normal samples obtained by the pretrained model in the memory library, and then judge whether the test samples are normal according to the distance between the features. The subimage anomaly detection with deep pyramid correspondences (SPADE) [25] combines features extracted from the pretrained network with the K-nearest neighbor (KNN) to obtain anomaly scores. Based on SPADE, Roth et al. [26] adjusted the feature extraction part of the model to achieve better performance.

Such methods are simple and practical, but their common problem is that the memory increases as the dataset increases and the features extracted by the pretrained model without any adjustment are affected by the ImageNet dataset [27]. In addition, noise or redundant information in the memory bank may reduce the accuracy and robustness of the detection. When the memory bank contains abnormal noise, even if the test sample is in the normal feature distribution range, it may still be misjudged due to the close distance from the outliers. Therefore, we expect the normal features to be more compact and separated from the anomaly features so that the unorganized features follow a certain distribution. In the existing work, the Gaussian distribution has been applied in anomaly detection. The Gaussian clustering of pretrained feature (GCPF) [28] estimated the multidimensional Gaussian distribution by calculating the mean and variance of features. In the work of the patch distribution modeling framework (PaDiM) [29], the feature map is divided into many small patches, and then the Gaussian distribution at each patch location is estimated, which improves the accuracy of distribution estimation but requires more time. Since existing work has demonstrated the effectiveness of the Gaussian distribution in anomaly detection, we use the Gaussian mixture model (GMM) to cluster the extracted features. In our proposed SMCC, the pretrained model is used to extract features. At the patch level, GMM clustering and sampling are used to obtain a central group representing each Gaussian distribution, and then the central group is used to supervise and guide the pretrained network and memory update. The purpose of using a self-updated memory library is to reduce the memory usage and adjust the parameters of the pretrained network to make it more suitable for the distribution of the current dataset. The purpose of combining the self-updated memory library with the GMM center clustering method is to avoid the memory size and noise problems and optimize the feature distribution.

Algorithm 1 Framework of the Proposed SMCC

Feature Extraction
Input: Training images set X, Pretrained model Φ
for x in X **do**
  $\phi = \Phi(x)$
  for j in $\mathcal{H}$ **do**
    concat $Resize(\phi_j)$ to ϕ_c
  end for
end for
ϕ_c is divided into $H \times W$ patches
Output: $\mathcal{P}$
GMM cluster
Input: $\mathcal{P}$, number of clusters K, sampling rate r
for x in X **do**
  Clustering with **GMM** and sampling
end for
Output: Center set S
Self-Updated Memory Bank
Input: Pretrained model Φ, patches $\mathcal{P}$, Center set S, epochs $\mathcal{W}$
Initialize memory bank $\mathcal{M}$
for i in $\mathcal{W}$ **do**
  for x in X **do**
    Calculate the distance between patches and center and update the distribution of patches
  end for
end for
return The updated $\mathcal{M}'$ and pretrained model Φ'

分割为多个补丁块

III. Proposed Anomaly Detection Method

The proposed framework SMCC is presented in Fig. 1. The SMCC has two stages and three modules. The two stages

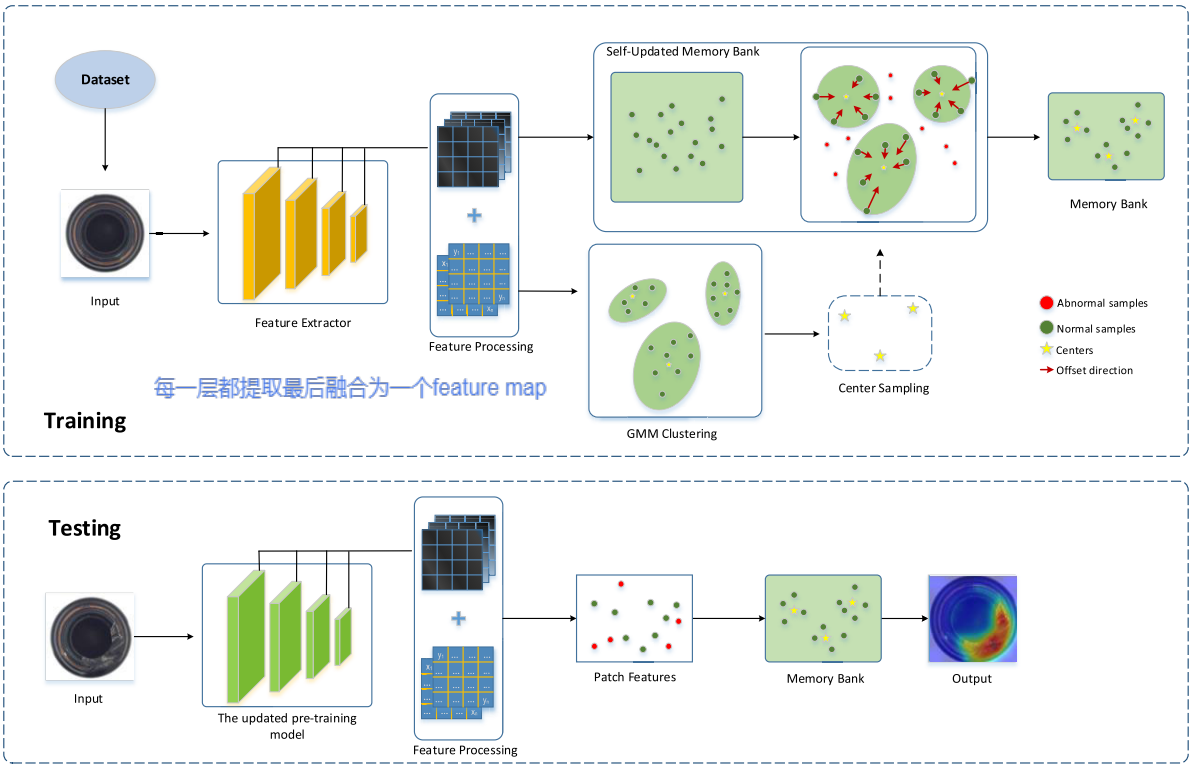

Fig. 1. Framework of the proposed SMCC.

are the training and testing stages. The three modules are the feature extraction and processing, the GMM clustering and center sampling, and the self-updated memory bank. Algorithm 1 shows the training steps of SMCC.

A. Feature Extraction and Processing

The models used for feature extraction are all pretrained on ImageNet. In this article, the feature extractor is denoted as Φ. The training dataset consists of photographs of industrial product surfaces without anomalies, while the test dataset contains images of normal and abnormal products. We use $\{x_1, \ldots, x_N \in X : l(x) = 0\}$ to denote the set of all the training images, where $l(x)$ indicates that the image is normal ($l(x) = 0$) or abnormal ($l(x) = 1$). Accordingly, we define $\{y_1, \ldots, y_N \in Y : l(y) \in \{0, 1\}\}$ to be the set of the test samples. The features in the product image include its edges, contours, colors, attributes, and other semantic information. The information of these features is extracted from the image by the feature extractor Φ. Many works [26], [28] have shown that the feature information extracted by each layer of the convolutional neural network (CNN) model is different. Specifically, the top layers in the network extract the underlying features, such as edge features; the latter layers in the network extract high-level features, which are the reorganization of low-level features and can highly summarize the attributes of the entity itself. Therefore, SMCC fuses the feature maps extracted from different layers to form a more comprehensive feature information. Each layer of the feature map output by the feature extractor is represented by $\phi_{i,j} = \Phi(x_i), i, j \in \mathbb{N}$, where i denotes the number of samples, and j represents the number of feature layers. With ResNet18 as an example, the four feature maps output by its

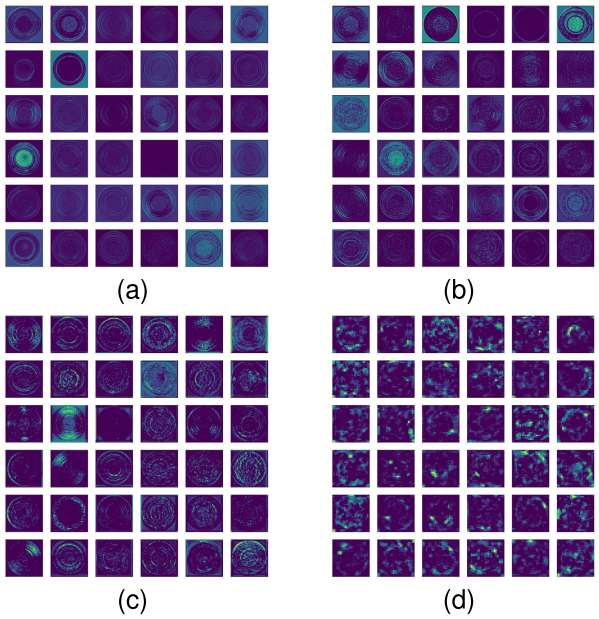

Fig. 2. (a)–(d) Feature map of ResNet18 layers one, two, three, and four output (by channel).

four layers are fused into a feature map ϕ_c through operations such as convolution and concat. The four feature maps output from the four layers of ResNet18 are shown in Fig. 2.

Then, SMCC divides the feature map into patches, so that the subsequent processing of features is at the patch level rather than the level of the entire feature map. Assuming that the feature map ϕ_i has a resolution of $H \times W$, if the size of each patch is 1×1, the feature map is divided into $H \times W$ patches $\mathcal{P}_{m,n}$ (m,n denotes the position) at most.

提取到的第i个数据第j层的特征

会不会分的有点多

B. GMM Clustering and Center Sampling

To solve the problem of noise described in Section II, SMCC clusters the feature patches of normal samples. The effectiveness of Gaussian clustering in anomaly detection has been confirmed by many works. Considering that a single Gaussian model is difficult to meet the complexity of different project features, we use a Gaussian mixture model to better fit the diverse feature distribution. GMM clusters the training samples and makes it away from the abnormal samples, while forming multiple clustering contours. We then construct a center set S of cluster centers that are expected to approximate the distribution of clusters and their central locations. Given a set S_o of input patches $\mathcal{P}$, a center set [30] is a weighted subset so that we can have a good approximation of the solution on the center set S and the original dataset S_o.

GMM can be regarded as a model composed of k single Gaussian submodels, and the submodel is the hidden variable of the mixture model. In theory, GMM can effectively fit the probability distribution of any shape. Now we assume the input data: $x_1, x_2, \ldots, x_N \subset \mathbb{R}^d$, GMM assumes that x_i is independent of each other and distributes according to the weighted average of k multivariate normal distributions, where k needs to be given in advance. The probability distribution of the Gaussian mixture model is

$$p(x_i \mid \theta) = \sum_{j=1}^{k} w_j \mathcal{N}\left(x_i \mid \mu_j, \Sigma_j\right) \tag{1}$$

where $\theta = (\mu_1, \Sigma_1, w_1, \ldots, \mu_k, \Sigma_k, w_k)$, which is the expectation, covariance, and probability of occurrence in the mixed model for each submodel. The mixture weights $w_j \in [0, 1]$ and $\sum_{j=1}^{k} w_j = 1$, and $\mathcal{N}(x; \mu_i, \Sigma_i)$ denote the multivariate normal distribution of the ith component of the mixture established by means and covariance parameters. It can be seen from the above formula that the key to complete the modeling of data probability density function is to estimate the parameter vector θ. The most common approach is to estimate θ via maximum likelihood estimation (MLE). For GMM, the log-likelihood function is

$$\mathcal{L}(\theta; X) = \sum_{i=1}^{N} \log p(x_i \mid \theta) \tag{2}$$

where N denotes the total number of datasets X. The maximization of the log-likelihood is accomplished via the expectation–maximization (EM) algorithm.

Our goal is to approximate the dataset $X = (x_1, \ldots, x_N)$ by a weighted set $S = \{(\gamma_1, \mathbf{x}'_1), \ldots, (\gamma_m, \mathbf{x}'_m)\} \subseteq \mathbb{R}_+ \times \mathbb{R}^d$ such that $\mathcal{L}(X \mid \theta) \approx \mathcal{L}(\mathcal{S} \mid \theta)$, where we define

$$\mathcal{L}(\theta; S, \gamma) = \sum_{i=1}^{Q} \gamma_i \log p\left(x'_i \mid \theta\right) \tag{3}$$

where Q denotes the total number of datasets S, and γ_i denotes the weight. The whole scheme of the center set sampling based on GMM is shown in Algorithm 2.

C. Self-Updated Memory Bank

The update of the memory bank is achieved by updating the pretrained model. In the previous work [12], the pretrained model is not updated, which makes the feature extraction affected by the pretrained dataset. In most of the work on supervised models [31], the pretrained model is updated by the label of the training set. In SMCC, clustering learning is performed first and then the pretrained model is updated through the clustering centers. The whole process does not require additional labels and training, so it is called self-update. In the self-updating process, the patches stored in the memory library are guided by the center set obtained by the previous clustering and enter into different clustering groups. As shown in Fig. 3(a), the center point attracts the surrounding patches, making them form multiple shapes around the center point. In the contours of these shapes, the normal sample cluster has multiple cores, and its shape is adaptively adjusted by training. Because only normal samples are trained, abnormal samples are naturally far away when normal $\mathcal{P}$ gradually enters each contour. To determine the escape route of a normal sample, we calculate its distance from each central point and then send it to the nearest central point.

Algorithm 2 Center Set Construction Algorithm

Input: Data set X, Number of clusters k
k-means++ was used to obtain the initial approximate solution
return bicriteria approximation $\mathcal{B}$,approximation factor α
Input: Dataset X, bicriteria approximation $\mathcal{B}$, approximation factor α, coreset size m.
for $j = 1$ to $|\mathcal{B}|$ **do**
$X_j \longleftarrow$ Set of points from X closest to point $\mathcal{B}_j$.
end for
for $j = 1$ to $|\mathcal{B}|$ **do**
$s(x) \leftarrow \alpha \mathrm{d}(x, \mathcal{B})^2 + \frac{2\alpha}{|\mathcal{X}_j|} \sum_{x' \in \mathcal{X}_j} \mathrm{d}(x', \mathcal{B})^2 + \frac{2}{|\mathcal{X}_j|} \sum_{x' \in \mathcal{X}} \mathrm{d}(x', \mathcal{B})^2$
end for
for x in X **do**
$q(x) \leftarrow \frac{s(x)}{\sum_{x' \in \mathcal{X}} s(x')}$
end for
Sample m weighted points from X, where each point x is sampled with probability $q(x)$ and assigned a weight $\frac{1}{m \cdot q(x)}$

Fitting GMM model on the center set
return Center set S

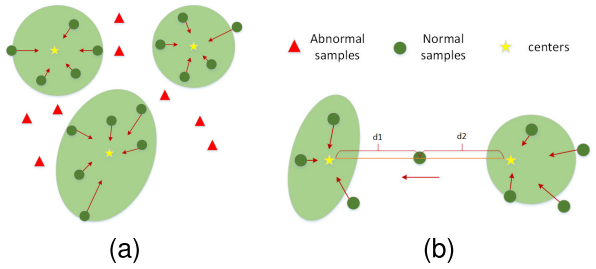

Fig. 3. Schematic of the center clustering process. (a) Process of patches being attracted center clustering. (b) Special case.

TABLE I
COMPLEXITY ESTIMATION OF DIFFERENT MODELS

Methods	Complexity of the model
SPADE [25]	$\mathcal{O}(H \times W \times C \times N)$
PaDiM [29]	$\mathcal{O}(H \times W \times C^2 \times N^2)$
SMCC	$\mathcal{O}(H \times W \times C \times B)$

This process is achieved by iterating to reduce the value of the loss function $\mathcal{L}_c$

$$\mathcal{L}_c = \sum_{i=1}^{M} \min\{(\mathcal{D}(\mathcal{P}_i, \mathcal{C}_n), n \in [1, N]\} \tag{4}$$

where $M = H \times W$ is the number of $\mathcal{P}$, and $\mathcal{D}(\mathcal{P}_i, \mathcal{C}_n)$ represents the calculation of the distance between each patch and different centers. However, it is not enough to use only one loss function. Because there could be a $\mathcal{P}$ that has more than one nearest center when $\mathcal{P}$ is located at the midpoint of two center points [Fig. 3(b)]. For this reason, we add a loss function $\mathcal{L}_o$ to supplement the lack of constraints. It calculates the distance d_1 of $\mathcal{P}_i$ to the first closest center C_1 and the distance d_2 to the second closest center C_2, and then subtracts them so that $\mathcal{P}_i$ moves closer to C_1 and further away from C_2

$$\mathcal{L}_o = \sum_{i=1}^{M} \mathcal{D}(\mathcal{P}_i, \mathcal{C}_1) - \mathcal{D}(\mathcal{P}_i, \mathcal{C}_2). \tag{5}$$

Finally, the loss function of SMCC $\mathcal{L}_{\text{SMCC}}$ consists of two parts. The memory bank is updated by continuously decreasing the loss function, and the pretrained parameters are also updated after each iteration

$$\mathcal{L}_{\text{SMCC}} = \mathcal{L}_c + \mathcal{L}_o. \tag{6}$$

SMCC saves the updated network parameters and memory bank and retains the best set of them after each training session. In each training of SMCC, the memory and feature extractor are updated, and only a fixed number of patches are saved, which makes the memory size of SMCC not affected by the number of samples. Table I records the complexity estimates of SMCC and previous related work, where H and W are the height and width of the feature map, respectively, and C is the number of channels of the feature map. N represents the total number of samples in the dataset, and B represents the sample size of one batchsize. Obviously, the complexity of SMCC is lower than that of SPADE and PaDiM.

D. Testing and Anomaly Score

During testing, the pretrained model Φ' obtained after training is used to extract feature information from the test images. Then these features are merged and divided into patches, as in the training phase. Finally, the anomaly score is calculated and used to determine whether the sample is abnormal. Anomaly score is obtained by calculating the distances between patches in the test image and patches in the memory bank. The formula for calculating the distance is as follows:

$$\mathcal{D} = \sqrt[2]{\sum_{i=1}^{P} \sum_{m=1}^{M} (x_i - p_m)^2} \tag{7}$$

TABLE II
TRAINING PARAMETERS AND EXPERIMENTAL ENVIRONMENT

Experimental environment	Parameter
GPU	NVIDIA A10
CPU	Intel(R) Xeon(R) Gold 5320
Operating system	Ubuntu18.04
Programming language	Python3.8
Deep learning framework	Pytorch1.10
Optimizer	Adam
Batch size	16
Sampling rate	0.1
Learning rate	0.001
Weight decay	0.0005

where M represents the number of patches in memory bank, and P represents the number of patches in extracted features. The anomaly score is the mean of the K nearest distances

$$\mathcal{S}_c = \frac{1}{K} \sum \min_k \mathcal{D}. \tag{8}$$

With the anomaly score of each position, the anomaly location of the image can be obtained. To match the original input resolution, we adjust the result image by bilinear interpolation. In addition, we use Gaussian smoothing to reduce noise and enhance image quality.

IV. EXPERIMENTS

A. Implementation Details and Evaluation Metric

Our experiments were conducted on the MVTec AD [32] dataset. The dataset contains 5354 images of industrial products in ten object categories (bottle, pill, capsule, hazelnut, metal nut, pill, screw, toothbrush, transistor, and zipper) and five texture categories (carpet, grid, leather, tile, and wood). The anomaly area of the dataset provides a pixelwise label with 70 different types of anomaly defects. Only normal images are provided by the dataset for training, while the test set contains normal and abnormal images. The more details of MVTec AD can be seen in [32]. Following the practice of previous studies, we tested each category separately.

Table II shows the setting of hyperparameters for training and the experimental environment. All the models used in this article are pretrained on the ImageNet dataset. And this article uses Wide_ResNet50 as the feature extractor, and the image size is fixed to 224 × 224 pixels. Unless specified, these parameters are used by default in the following experiments. All the experimental results in this article are the average of at least three repeated experiments.

To get an additional performance measure that is independent of the determined threshold, we use image-level and pixel-level area under the receiver operating characteristic curves (AUROCs) as the criteria for model evaluation. In general, image-level AUROC is used to evaluate the ability of anomaly detection; pixel-level AUROC is used to evaluate the ability of anomaly localization. However, the problem of pixel-level AUROC as an evaluation metric for anomaly location is: a correctly segmented large region can make up for many incorrectly segmented small regions, which may affect the accuracy of the results to some extent. Therefore, to more accurately evaluate the anomaly location of the model,

TABLE III
PERFORMANCE OF DIFFERENT PRETRAINED MODELS

Backbone	ResNet18	ResNet50	WRN50	ResNet101	WRN101
I-AUROC (%)	97.2	97.8	98.5	97.9	**98.6**
P-AUROC (%)	97.9	98.0	98.2	98.0	**98.3**
PRO (%)	93.0	93.0	**93.1**	93.0	**93.1**
Parameters (M)	**1.8**	5.2	25.6	26.3	50.1

per-region-overlap (PRO) [33] is used as a supplementary evaluation metric. It weights ground-truth regions of different sizes equally so it can be used as a supplement to pixel-level AUROC. The model's performance in anomaly detection is measured by image-level AUROC, and its performance in anomaly localization is measured by pixel-level AUROC and PRO.

B. Influence of Pretrained Models

Pretrained model is an application of transfer learning. The pretrained model and fine-tuning mechanism have good scalability. When supporting new tasks, the data only need to be fine-tuned. Our experiments tried various pretrained backbone models as feature extractors. First, ResNet18, ResNet50, and ResNet101 [34] were studied, and then Wide_ResNet50 (WRN50) and Wide_ResNet101 (WRN101) [35] were studied. The results of experiments are reported in Table III.

It can be observed from the table that the depth and width of the backbone have some influence on the performance of the model. From the data, the performance of ResNet50 is significantly better than ResNet18, which means that deeper models have better learning ability and fit more complex feature inputs to have stronger expressiveness. But the depth has little effect after it reaches 50 layers, which means that 50 layers of learning are enough for our model. By observing WRN50 and WRN101, it can be seen that their performance is significantly better than the corresponding ResNet. In addition, it is worth noting that the increase in width is more obvious for the pixel-level AUROC, and the depth has almost no effect on the pixel-level AUROC. This is because an increase in width means an increase in the number of channels, which allows each layer of the network to extract more features. As shown in Fig. 2, we visualize the feature information extracted from the different layer of the pretrained model channel by channel. Obviously, the information stored in different channels is different, some focus on edge information, some focus on texture information, some focus on global information and so on. For the anomaly detection task, global information is emphasized in feature extraction, while more details and complex texture information are needed in the anomaly location task.

图像级变化的多

To further study the impact of different layers of the pretrained network on performance, we tested the models using different network layers. The results are recorded in Fig. 4. Obviously, the effect of using a certain layer of the network alone is not ideal. The model that combines the features of one, two, three, and four layers has the best effect. Therefore, our method combines the features of all the layers.

C. Influence of Batch Size and Sampling

The batch size and sampling represent the number of patches and cluster centers in the memory bank. The memory

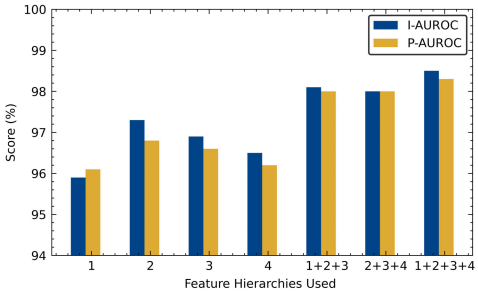

Fig. 4. Network feature depths on anomaly detection performance.

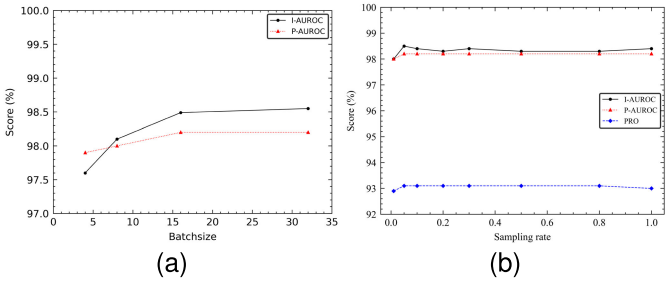

Fig. 5. Batchsize and different sampling rates on performance. (a) Results of the model using different batchsize. (b) Results of the model using different sampling rate.

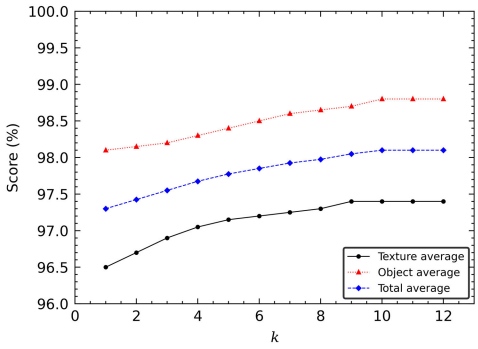

Fig. 6. Impact of different k values on performance.

bank stores a batch of patches for each update, so batch size determines the size of the memory bank. We explored the effect of memory size on SMCC performance by changing the batch size. The experimental results are shown in Fig. 5(a). It can be seen that the larger the batch, the better the model performance. However, when the batch is greater than 16, the performance of the model does not change much, which indicates that our model does not depend on the storage of a large number of features. Then, we focus on the change in cluster centers due to different sampling rates, which determines how many centers are there in the update process. We explored this problem by setting different sampling rates, and the results are shown in Fig. 6(b). The results show that when the sampling rate changes, the indicators obtained by the model are almost unchanged. This shows that SMCC does not need too many clustering centers to achieve performance saturation, which is conducive to saving computing resources.

D. Influence of Parameter k

In GMM clustering, the number of clusters k plays an important role. However, due to the complexity of the

TABLE IV
DIFFERENCES IN MODEL PERFORMANCE WHEN THERE ARE ONLY ONE, TWO, OR NO LOSS FUNCTIONS

Backbone	$\mathcal{L}_c$	$\mathcal{L}_o$	I-AUROC (%)	P-AUROC (%)
ResNet18			95.5	97.3
	√		97.1	97.9
	√	√	97.2	97.9
WRN50			96.3	97.5
	√		98.3	**98.2**
	√	√	**98.5**	**98.2**

evaluated items, it is difficult to theoretically analyze the role of k. We observed its effect on SMCC performance by setting different k values. As shown in Fig. 6, when the k value is less than 4, the gain of the texture class is greater than that of the object class. This can be explained as that the factors of texture class are relatively simple, so only less clustering can meet the needs. When k is greater than 10, the change in k has little effect on performance, which indicates that the number of clusters has met the demand to a certain extent. In general, the gain is greater when it is less than 6, which means that a few clusters cannot fully express complex prototypes. An increase in the k value will result in a certain gain in the performance, but it will also increase the memory. The k value we selected in the experiment is 10, and the application needs to be determined according to the actual situation.

E. Influence of Loss Function

To further verify the validity of the self-updated memory bank, we set up the following experiments to explore the effects of the two loss functions of SMCC. Table IV records the results of SMCC using one, two, or no loss functions in anomaly detection. No loss function means that the self-updated module does not take effect.

Obviously, $\mathcal{L}_c$ has the greatest influence on the results, while $\mathcal{L}_o$ only plays an auxiliary role, and the results are consistent with our theoretical analysis. When there is no loss function, features are randomly distributed, which is similar to SPADE, where features are simply stored without any arrangement, so the performance is naturally worse. When the loss function $\mathcal{L}_c$ is added, the distribution of features will converge to the Gaussian center and follow its classes, so the performance of the model becomes better. However, when a feature point falls in the middle of two or more taxa, it will be confused about which taxa it belongs to, and the appearance of $\mathcal{L}_o$ is just to solve this problem. After all, this is a minority of cases, so the performance gain of $\mathcal{L}_o$ is not particularly obvious. From the overall results, the loss function has gain for both image-level and pixel-level AUROCs, which means that the self-updated memory bank module improves the accuracy of the model.

F. Anomaly Detection on MVTec AD

In this section of the experiment, some test result graphs are preserved to more intuitively show the performance of our model. The parameters used in the model are consistent with those described in Table II, and the pretrained model used is Wide_ResNet50. Fig. 7 shows some results of SMCC detection on the MVTec AD dataset. It shows the original

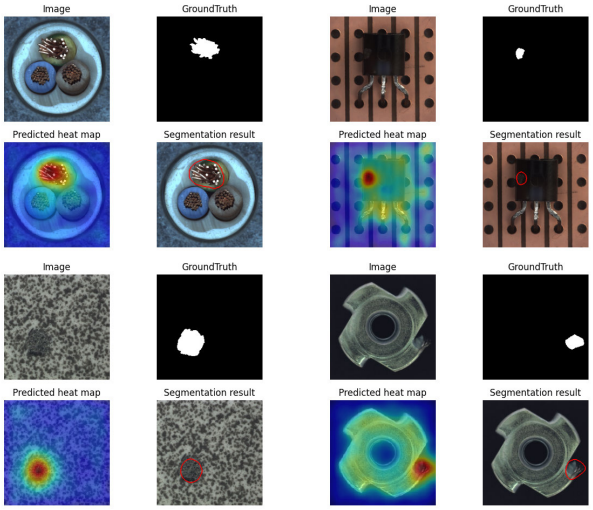

Fig. 7. Visual results of anomaly detection and location for some products (cable, transistor, tile, metal nut).

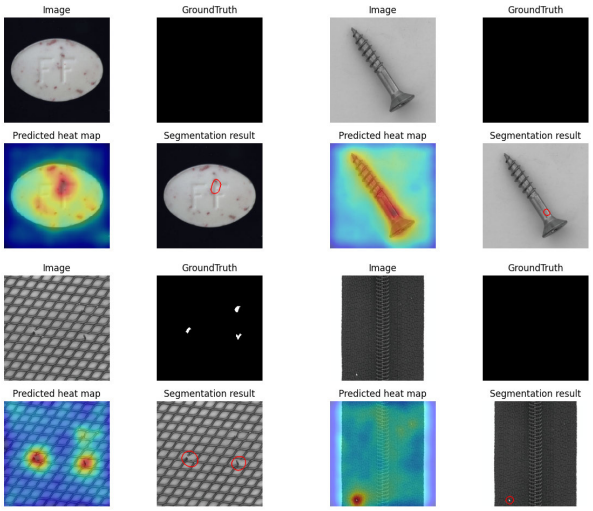

Fig. 8. Some failure cases (pill, screw, grid, zipper).

images, the ground-truth images, the heat map predicted by our model, and the anomaly segmentation result diagram. The white part in the ground-truth images reflects the defective part of the product, and the complete black indicates that the product is free of defects. Overall, SMCC can accurately detect and locate abnormal parts in the image. However, there are still a few failure cases, which mainly include two types: the missing detection cases and the false detection cases. Fig. 8 shows some failure cases. SMCC missed a small anomaly in the image of grid. The other three items in the picture were mis-detected. In all the failure cases, most are false positives. And it can be found that the possibility of missed detection in texture items is relatively large. This can be explained by the fact that SMCC treats some anomalies as texture changes. Mis-detections are often caused by the SMCC mistaking details of color changes (such as reflection points on metal objects) for anomalies. Finally, it can be concluded that the weakness of SMCC lies in its understanding of the details of texture and color changes.

TABLE V
PERFORMANCE COMPARISON OF IMAGE-LEVEL AUROC (%) ON EACH CLASS OF MVTec AD DATASET. BEST RESULTS ARE SHOWN IN BOLD

Category/Method	PatchSVDD [24]	SPADE [25]	PaDiM [29]	L^2AE-grad [14]	STPM [16]	GCPF [28]	CFlow [19]	SMCC
Carpet	92.9	98.6	-	73.4	-	95.3	**100.0**	**100.0**
Grid	94.6	99.0	-	98.1	-	56.5	97.6	**99.3**
Leather	90.9	99.5	-	92.1	-	99.7	97.7	**100.0**
Tile	97.8	89.8	-	57.5	-	99.0	98.7	**100.0**
Wood	96.5	95.8	-	80.5	-	97.3	**99.6**	**99.6**
Textures Average	94.5	96.5	99.0	80.3	-	89.6	98.7	**99.8**
Bottle	98.6	98.1	-	91.6	-	98.3	**100.0**	**100.0**
Cable	90.3	93.2	-	86.4	-	89.5	**100.0**	96.6
Capsule	76.7	98.6	-	95.2	-	94.5	**99.3**	96.6
Hazelnut	92.0	98.0	-	98.4	-	96.9	96.8	**100.0**
Metal Nut	94.0	96.9	-	89.9	-	87.0	91.9	**99.6**
Pill	86.1	96.5	-	91.2	-	86.4	**99.9**	96.5
Screw	81.3	99.5	-	98.0	-	73.1	**99.7**	91.1
Toothbrush	**100.0**	98.9	-	98.3	-	97.2	91.9	**100.0**
Transistor	91.5	81.0	-	92.1	-	91.6	99.1	**100.0**
Zipper	97.9	98.8	-	88.9	-	97.0	**98.5**	**98.5**
Objects Average	90.8	96.0	97.2	93.0	-	91.2	**98.0**	97.9
Total Average	92.1	96.2	97.9	88.8	95.5	90.3	98.3	**98.5**

TABLE VI
PERFORMANCE COMPARISON OF PIXEL-LEVEL AUROC (%) AND PRO (%) ON EACH CLASS OF MVTec AD DATASET. BEST RESULTS ARE SHOWN IN BOLD

Category/Method	PatchSVDD [24]	SPADE [25]	PaDiM [29]	DFR [15]	STPM [16]	GCPF [28]	SMCC
Carpet	92.6/-	97.5/94.7	**99.1/96.2**	97.0/93.0	98.8/95.8	98.9/-	99.1/95.6
Grid	96.2/-	93.7/86.7	97.3/94.6	98.1/92.9	**99.0/96.6**	97.8/-	97.7/93.6
Leather	97.4/-	97.6/97.2	99.2/97.8	98.0/96.9	**99.3/98.0**	**99.3**/-	**99.3**/96.8
Tile	91.4/-	87.4/75.9	94.1/86.0	87.1/79.9	**97.4/92.1**	96.1/-	96.2/87.4
Wood	90.8/-	88.5/87.4	94.9/91.1	93.0/91.1	**97.2/93.6**	95.1/-	94.5/89.7
Textures average	93.7/-	92.9/88.4	96.9/93.2	94.6/90.6	**98.3/95.2**	97.4/-	97.4/92.6
Bottle	98.1/-	98.4/95.5	98.3/94.8	96.9/93.0	**98.8**/95.1	97.5/-	**98.8/95.6**
Cable	96.8/-	97.2/90.9	96.7/88.8	92.0/81.1	95.5/87.7	95.7/-	**98.6/91.5**
Capsule	95.8/-	99.0/93.7	98.5/93.5	99.0/93.9	98.3/92.2	97.7/-	**99.1/94.2**
Hazelnut	97.5/-	**99.1**/95.4	98.2/92.6	99.0/95.8	98.5/94.3	98.1/-	98.9/**96.0**
Metal Nut	98.0/-	98.1/94.4	97.2/85.6	93.1/92.1	97.6/94.5	95.9/-	**99.3/95.2**
Pill	95.1/-	96.5/94.6	95.7/92.7	97.2/96.2	97.8/96.5	97.0/-	**98.8/96.6**
Screw	95.7/-	98.9/96.0	98.5/94.4	**99.1/96.1**	98.3/93.0	97.5/-	98.3/93.0
Toothbrush	98.1/-	97.9/93.5	97.5/93.1	99.0/**93.9**	98.9/92.2	97.2/-	**99.1**/91.6
Transistor	97.0/-	94.1/87.4	**98.5**/84.5	80.1/79.9	82.5/69.5	90.7/-	97.7/**94.8**
Zipper	95.1/-	96.5/92.6	97.8/**95.9**	96.0/90.8	98.5/95.2	98.2/-	**98.9**/95.7
Objects average	96.7/-	97.6/93.4	97.6/91.6	95.1/91.2	96.5/91.0	96.6/-	**98.8/94.4**
Total average	95.7/-	96.5/91.7	97.5/92.1	95.1/91.1	97.0/92.4	96.8/-	**98.3/93.8**

G. Comparison With the State-of-the-Art Methods

Tables V and VI record the comparison experiments of our method with some other recently proposed advanced methods on the MVTec AD dataset. Table V records the image-level AUROC score, which reflects the model's performance of the anomaly detection. Table VI records the scores for the pixel-level AUROC and PRO, which reflects the model's performance of the anomaly localization. The methods used here include: Patch-SVDD [24], SPADE [25], PaDiM [29], the variant variational autoencoder model L^2 AE-grad [14], STPM [16], GCPF [28], CFlow [19], DFR [15], and ours.

The following observations are made from the data in the tables.

1) SMCC has a better performance than SPADE whether it is anomaly location or detection. SPADE directly uses the features extracted by the pretrained model to classify by the KNN method. Differently, SMCC clusters the extracted features to avoid the classification results being affected by noise and updates the pretrained model through the self-renewal module, which makes the parameters of the pretrained model more suitable for the current task.
2) GCPF performs Gaussian clustering on the features extracted from each layer of the pretrained model and then integrates the results, while PaDiM divides the feature map into many patches to learn Gaussian distribution, respectively. As a result, PaDiM's processing of features is more refined, which is one of the reasons why PaDiM performs better than GCPF. In comparison, SMCC not only makes full use of each layer of information but also divides the feature map into patches for clustering. More importantly, the pretrained model is updated by the self-update module, which makes the performance of SMCC surpass GCPF and PaDiM.
3) CFLow can learn complex feature representations through the powerful fitting ability of NF, which makes such methods have good performance. In contrast,

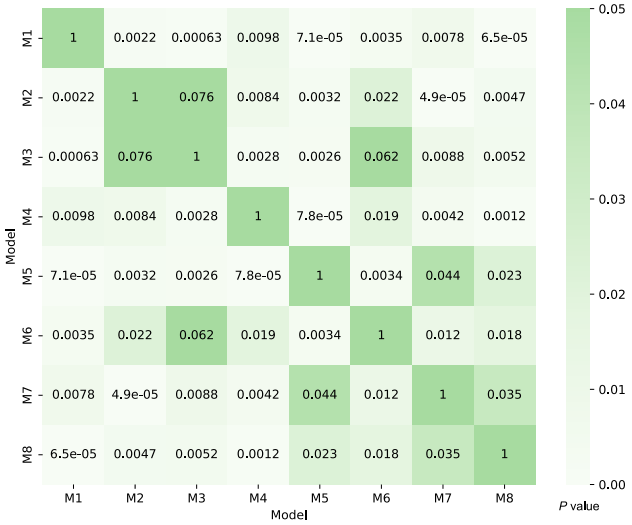

Fig. 9. Heat map of significant difference.

TABLE VII
PERFORMANCE OF SEVERAL METHODS WHEN USING THE SAME BACKBONE (WRN50)

Methods	Infer. Time(s)	Memory(M)	Performance(%)
SPADE (WRN50)	0.66	1400	85.5/96.5/91.7
PaDiM (WRN50)	0.19	3800	95.5/97.5/92.1
SMCC (WRN50)	**0.12**	**344**	**98.5/98.2/93.1**

SMCC can perform at the same level or even better with less training.

4) As shown in Table V, the overall performance of SMCC in the anomaly detection tasks is outstanding. From Table VI, it can be found that SMCC is relatively weak in anomaly localization of texture classes. We will consider solving this problem in the future by improving the clustering methods and enhancing the feature extraction (such as obtaining more detailed information through a wider model).

To evaluate the performance of the model more scientifically, we conducted a statistical significance test on the experimental data of different models. Because our samples are random and independent, and based on the central limit theorem, our data are suitable for paired t-test. Fig. 9 records the results of our paired t-tests between different models, where M1–M8 represent the eight models in Table V. In the heat map, if the P value is less than 0.05, it indicates that the significant difference between the two models is relatively large, and the two models are very different. Obviously, SMCC and other models have large difference values, which means that SMCC does have outperforming performance.

For an anomaly detection model, its memory size and inference time are also our focus. Table VII reports the inference time and memory size of several different models. The pretrained model used in the experiment is WRN50, as shown in the table. Obviously, SMCC outperforms SPADE and PaDiM when using the same backbone. While SMCC is higher than the other two models in the three indicators of anomaly detection, it occupies less memory and uses less inference time.

TABLE VIII
MEAN ANOMALY LOCALIZATION PERFORMANCE ON BTAD AND DAGM

BTAD	L^2AE-grad [14]	SPADE [25]	VT-ADL [36]	SMCC
P-AUROC (%)	78.2	79.3	90.2	**96.6**
DAGM	L^2AE-grad [14]	SPADE [25]	GCPF [28]	SMCC
P-AUROC (%)	81.3	94.8	96.6	**97.1**

H. Anomaly Detection on Other Datasets

To further evaluate the general applicability of SMCC, we conducted additional experiments on the beanTech Anomaly Detection (BTAD) [36] and Deutsche Arbeitsgemeinschaft für Mustererkennung e.V. and German chapter of the International Association for Pattern Recognition (DAGM) [37] datasets. The BTAD dataset contains a total of 2540 images of three categories of industrial products. The DAGM dataset was published by the German Association for Pattern Recognition. The dataset consists of ten different texture categories. Various anomalies occur on various texture backgrounds in the dataset. The training set of these datasets contains only normal images, while the test set contains normal and abnormal images. The more details of BTAD and DAGM can be seen in [36] and [37]. Table VIII records the average anomaly localization performance of different methods on BTAD and DAGM. Here, we choose several representative methods and SPADE which is closely related to our method. VT-ADL is an anomaly detection method proposed by the author of the BTAD dataset. It is obvious that our SMCC performs better than other methods.

V. CONCLUSION

We propose a new unsupervised framework SMCC for anomaly detection and location. In the test of the MVTec AD dataset, the image-level AUROC reflecting the ability of anomaly detection reached 98.5%, and the pixel-level AUPROC and PRO reflecting the ability of anomaly location reached 98.3% and 93.8%, respectively. On the BTAD and DAGM datasets, the pixel-level AUROC of SMCC reaches 96.6% and 97.1%, respectively. Besides, our experiments show that SMCC consumes less memory than other similar methods. Through experimental analysis, we find that the weakness of SMCC is the anomaly localization for texture changes and subtle color changes. In the future, we will focus on the anomaly localization of complex texture and color classes and continue to optimize the speed and accuracy of our models.

REFERENCES

[1] Y. He, K. Song, Q. Meng, and Y. Yan, "An end-to-end steel surface defect detection approach via fusing multiple hierarchical features," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 4, pp. 1493–1504, Apr. 2020.
[2] Y.-X. Zhang, S.-W. Han, J. Cheon, S.-J. Na, and X.-D. Gao, "Effect of joint gap on bead formation in laser butt welding of stainless steel," *J. Mater. Process. Technol.*, vol. 249, pp. 274–284, Nov. 2017.
[3] X. Gao and Y. Zhang, "Monitoring of welding status by molten pool morphology during high-power disk laser welding," *Optik*, vol. 126, pp. 1797–1802, Oct. 2015.
[4] T. Wang, X. Gao, K. Seiji, and X. Jin, "Study of dynamic features of surface plasma in high-power disk laser welding," *Plasma Sci. Technol.*, vol. 14, no. 3, pp. 245–251, Mar. 2012.
[5] X. Gao, Y. Sun, and S. Katayama, "Neural network of plume and spatter for monitoring high-power disk laser welding," *Int. J. Precis. Eng. Manufacturing-Green Technol.*, vol. 1, no. 4, pp. 293–298, Oct. 2014.

[6] Y. Feng et al., "Simulation and experiment for dynamics of laser welding keyhole and molten pool at different penetration status," *Int. J. Adv. Manuf. Technol.*, vol. 112, nos. 7–8, pp. 2301–2312, Feb. 2021.

[7] X. Gao, D. Ding, T. Bai, and S. Katayama, "Weld-pool image centroid algorithm for seam-tracking vision model in arc-welding process," *IET Image Process.*, vol. 5, no. 5, pp. 410–419, 2011.

[8] X. Gao, L. Mo, D. You, and Z. Li, "Tight butt joint weld detection based on optical flow and particle filtering of magneto-optical imaging," *Mech. Syst., Signal Process.*, vol. 96, pp. 16–30, Nov. 2017.

[9] X. Fan, X. Gao, G. Liu, N. Ma, and Y. Zhang, "Research and prospect of welding monitoring technology based on machine vision," *Int. J. Adv. Manuf. Technol.*, vol. 115, nos. 11–12, pp. 3365–3391, Aug. 2021.

[10] J. Luo, Z. Yang, S. Li, and Y. Wu, "FPCB surface defect detection: A decoupled two-stage object detection framework," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–11, 2021.

[11] Z. Tu, S. Wu, G. Kang, and J. Lin, "Real-time defect detection of track components: Considering class imbalance and subtle difference between classes," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–12, 2021.

[12] X. Tao, X. Gong, X. Zhang, S. Yan, and C. Adak, "Deep learning for unsupervised anomaly localization in industrial images: A survey," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–21, 2022.

[13] Y. Yan, D. Wang, G. Zhou, and Q. Chen, "Unsupervised anomaly segmentation via multilevel image reconstruction and adaptive attention-level transition," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–12, 2021.

[14] D. Dehaene, O. Frigo, S. Combrexelle, and P. Eline, "Iterative energy-based projection on a normal data manifold for anomaly localization," 2020, *arXiv:2002.03734*.

[15] Y. Shi, J. Yang, and Z. Qi, "Unsupervised anomaly segmentation via deep feature reconstruction," *Neurocomputing*, vol. 424, pp. 9–22, Feb. 2021.

[16] G. Wang, S. Han, E. Ding, and D. Huang, "Student-teacher feature pyramid matching for anomaly detection," 2021, *arXiv:2103.04257*.

[17] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1530–1538.

[18] J. Yu et al., "FastFlow: Unsupervised anomaly detection and localization via 2D normalizing flows," 2021, *arXiv:2111.07677*.

[19] D. Gudovskiy, S. Ishizaka, and K. Kozuka, "CFLOW-AD: Real-time unsupervised anomaly detection with localization via conditional normalizing flows," 2021, *arXiv:2107.12571*.

[20] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[21] J. Chen, T. Wang, X. Gao, and L. Wei, "Real-time monitoring of high-power disk laser welding based on support vector machine," *Comput. Ind.*, vol. 94, pp. 75–81, Jan. 2018.

[22] W. Mao, J. Chen, X. Liang, and X. Zhang, "A new online detection approach for rolling bearing incipient fault via self-adaptive deep feature matching," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 2, pp. 443–456, Feb. 2020.

[23] K. Sohn, C.-L. Li, J. Yoon, M. Jin, and T. Pfister, "Learning and evaluating representations for deep one-class classification," 2020, *arXiv:2011.02578*.

[24] J. Yi and S. Yoon, "Patch SVDD: Patch-level SVDD for anomaly detection and segmentation," 2020, *arXiv:2006.16067*.

[25] N. Cohen and Y. Hoshen, "Sub-image anomaly detection with deep pyramid correspondences," 2020, *arXiv:2005.02357*.

[26] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, and P. Gehler, "Towards total recall in industrial anomaly detection," 2021, *arXiv:2106.08265*.

[27] S. Lee, S. Lee, and B. Cheol Song, "CFA: Coupled-hypersphere-based feature adaptation for target-oriented anomaly localization," 2022, *arXiv:2206.04325*.

[28] Q. Wan, L. Gao, X. Li, and L. Wen, "Industrial image anomaly localization based on Gaussian clustering of pretrained feature," *IEEE Trans. Ind. Electron.*, vol. 69, no. 6, pp. 6182–6192, Jun. 2022.

[29] T. Defard, A. Setkov, A. Loesch, and R. Audigier, "PaDiM: A patch distribution modeling framework for anomaly detection and localization," 2020, *arXiv:2011.08785*.

[30] M. Lucic, M. Faulkner, A. Krause, and D. Feldman, "Training Gaussian mixture models at scale via coresets," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 5885–5909, 2017.

[31] A.-A. Tulbure, A.-A. Tulbure, and E.-H. Dulf, "A review on modern defect detection models using DCNNs—Deep convolutional neural networks," *J. Adv. Res.*, vol. 35, pp. 33–48, Jan. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2090123221000643

[32] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, "MVTec AD—A comprehensive real-world dataset for unsupervised anomaly detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9584–9592.

[33] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, "Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 4182–4191.

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Mar. 2016, pp. 770–778.

[35] S. Zagoruyko and N. Komodakis, "Wide residual networks," 2016, *arXiv:1605.07146*.

[36] P. Mishra, R. Verk, D. Fornasier, C. Piciarelli, and G. L. Foresti, "VT-ADL: A vision transformer network for image anomaly detection and localization," in *Proc. IEEE 30th Int. Symp. Ind. Electron. (ISIE)*, Jun. 2021, pp. 1–6.

[37] M. Wieler and T. Hahn. (2017). *Weakly Supervised Learning for Industrial Optical Inspection*. [Online]. Available: https://hci.iwr.uni-heidelberg.de/node/3616

Yongheng Liu received the B.S. degree in mechanical engineering from Xiangtan University, Xiangtan, Hunan, China, in 2020. He is currently pursuing the M.S. degree in mechanical engineering from the Guangdong University of Technology, Guangzhou, Guangdong, China.

His current research interests include anomaly detection and deep learning.

Xiangdong Gao received the B.E. degree in automation from Zhengzhou University, Zhengzhou, China, in 1985, the M.A. degree in automation from Central South University, Changsha, China, in 1988, and the Ph.D. degree in welding from the South China University of Technology, Guangzhou, China, in 1998.

He is currently a Professor and the Director of Guangdong Provincial Welding Engineering Technology Research Center, Guangdong University of Technology, Guangzhou. His research interests include welding automation and machine vision.

James Zhiqing Wen received the Ph.D. degree in optical instruments from Tsinghua University, Beijing, China, in 1994.

From 1994 to 1995, he was a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, University of California at Santa Barbara, Santa Barbara, CA, USA. He worked as a Principal Scientist/Director of Research and Development in high-tech companies in USA for more than 20 years. He is currently working as the Executive Deputy Director with the Engineering Research Center for Intelligent Robotics, Ji Hua Laboratory, Foshan, China. He has authored and coauthored about 60 technical papers and more than 30 patents. His research interests include intelligent machine vision, pattern recognition, 3-D imaging, optical engineering, artificial intelligence (AI), virtual reality (VR)/augmented reality (AR), metaverse, intelligent robotics, and unmanned systems.

Dr. Wen received the 2002 Rudolf Kingslake Medal of the International Society of Optical Engineers.

Huiyuan Luo received the B.S. degree from the Harbin Institute of Technology, Weihai, China, in 2016, and the Ph.D. degree from the Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Science, Changchun, China, in 2021.

From 2021 to 2022, he was an Associate Researcher with the Ji Hua Laboratory, Foshan, China. He has been engaged in saliency detection, industrial anomaly detection, unsupervised learning, and intelligent manufacturing.