IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 7, 2021, accepted October 21, 2021, date of publication November 1, 2021, date of current version November 8, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3124525

Self-Supervised Learning for Anomaly Detection With Dynamic Local Augmentation

SEUNGDONG YOA, SEUNGJUN LEE, CHIYOON KIM, AND HYUNWOO J. KIM
Department of Computer Science, Korea University, Seoul 02841, Republic of Korea

Corresponding author: Hyunwoo J. Kim (hyunwoojkim@korea.ac.kr)

This work was supported in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) through the ICT Creative Consilience Program under Grant IITP-2021-2020-0-01819 funded by the Ministry of Science and ICT (MSIT), Government of Korea (Development of Integrated Cognitive Drone AI for Disaster/Emergency Situations), under Grant 2021-0-00025, in part by the Korea Institute of Planning and Evaluation for Technology in Food, Agriculture and Forestry (IPET) and the Korea Smart Farm Research and Development Foundation (KosFarm) through Smart Farm Innovation Technology Development Program, funded by the Ministry of Agriculture, Food and Rural Affairs (MAFRA) and the Ministry of Science and ICT (MSIT), in part by the Rural Development Administration (RDA) under Grant 421025-04, and in part by Samsung Electronics Company Ltd.

ABSTRACT Anomaly detection is an important problem for recent advances in machine learning. To this end, many attempts have emerged to detect unknown anomalies of the images by learning representations and designing score functions. In this paper, we propose a simple yet effective framework for unsupervised anomaly detection using self-supervised learning. We extend conventional self-supervised learning for an anomaly detection problem. In anomaly detection, anomalous patterns appear in the local regions of an image, so we employ *dynamic local augmentation* to generate a negative pair of the images from the normal training dataset. Specifically, in addition to learning the global representation of an image, our framework contrasts a normal sample to a locally augmented sample. To effectively apply the local augmentations regardless of a category or a random location of an image, we use dynamically weighted local augmentations to generate more suitable negative samples. We also present a novel scoring function for detecting unseen anomalous patterns. Our experiment demonstrates the effectiveness of our method, and we show that our framework achieves competitive performance compared to state-of-the-art methods on MVTec Anomaly Detection dataset.

INDEX TERMS Anomaly detection, computer vision, deep learning, machine learning, self-supervised learning.

I. INTRODUCTION

Anomaly detection, referred to as novelty detection, outlier detection, out-of-distribution (OOD), is a task that identifies abnormal, novel, or unseen data. In general, one of the assumptions in anomaly detection is that anomalies rarely occur and anomaly patterns are not given in the training dataset. Since anomaly data is not accessible during the training, many approaches have been proposed to detect anomalies by the unsupervised method.

Recently, there have been many approaches for image-level anomaly detection. For example, One-class classifiers [1], [2] have shown competitive results on image-level classification with high-level representations. However, pixel-level anomaly detection is a task to be more practical and precise in many domains of computer vision. Compared to image-level features, pixel-level features (dense features) are difficult to train due to the preservation of spatial features and the construction of dense outputs. By studying pixel-level anomalies, we can improve the way to extract the dense features in self-supervised learning. In practical use cases, pixel-level anomaly detection can perform well on vision inspection in industrial environments (e.g., CCTV) where inspection is done by multiple objects not by a single object. In that case, we expect that the pixel-level anomaly detection algorithm can better detect than the image-level algorithm.

To this end, previous works have been proposed based on Generative Adversarial Network (GAN) [3]–[5], Auto-Encoder (AE) [6]–[8]. GAN-based models detect anomalies when the confidence score is below a certain threshold. AE-based models are trained to reconstruct the input data which is normal, and determine anomalies when

The associate editor coordinating the review of this manuscript and approving it for publication was Sudipta Roy.

reconstruction error is high at test time. Since both methods consider pixel-wise reconstruction error, the methods lack to capture the high-level features and semantic information [9], [10].

In recent years, self-supervised learning approaches using contrastive learning have been successful in learning representations. Contrastive learning [11]–[14] also showed a strong ability to learn good representation in pixel-level representation tasks (e.g., object detection, semantic segmentation). A few works [15] have shown the effectiveness of contrastive learning in image-level anomaly detection. However, not many works have been proposed for pixel-level anomaly detection which requires accurate and reliable pixel-level representation.

Our work tackles pixel-level anomaly detection which segments anomaly pixels by self-supervised learning approaches. Self-supervised learning with only positive pairs [14], [16] has resulted in competitive performance in comparison to contrastive learning which requires both negative pairs and positive pairs [13], [17]. Since the training dataset consists of only normal data, our work extends the self-supervised learning using positive pairs with *dynamic local augmentation* to disassociate anomaly pixels. We also propose two novel scoring functions in terms of data and the model's layer: Regression error and Uncertainty per pixel. For computing the score of a pixel, we measure the difference of the maps (feature map or variance map) between the train images and the test image for each layer. Also, to generate the variance map for the uncertainty score, we use Monte Carlo Dropout to measure the uncertainty of the pixel representation.

The goal of the proposed method is to segment anomalies by self-supervised learning with dynamic local augmentation. Our **contributions** are as follows:

- We propose a self-supervised learning framework for anomaly detection with a novel augmentation method called *dynamic local augmentation* that is adaptively applied to each individual image locally to generate more delicate anomalies.
- We introduce two scoring functions that discriminate the anomalies from normal pixel-level features using regression errors and variance maps: The regression error is the difference between the average feature map of train images and the feature map of a test image. The other is the difference between the average variance map of train images and the variance map of the test image using Monte Carlo dropout for each layer (three last layers from each residual stage on ResNet-18).
- Lastly, our experiments demonstrate that the proposed method which is trained from scratch achieves the competitive performance on the public benchmark MVTec Anomaly Detection dataset.

The remainder of this paper is as follows. Related works on anomaly detection, self-supervised learning, uncertainty are discussed in Section II. Section III introduces our self-supervised learning framework on anomaly detection with two scoring function regression error and variance map. In Section IV, we discuss the pixel-level anomaly detection experiment results on MVTec dataset. Finally, Section V draws the conclusion and future work of our research.

II. RELATED WORKS

A. ANOMALY DETECTION

A wide range of literature on anomaly detection [18]–[26] has appeared in machine learning. In most anomaly detection tasks, we assume that only the normal data is given during the training and the model predicts whether a test sample is normal or not during the test time. A variety of papers for anomaly detection based on Generative models have been studied. These models such as Auto Encoder (AE) [6]–[8], [27] and Generative Adversarial Network (GAN) [3]–[5], [28], [29] are trained to generate normal data on the training dataset which includes only the normal data. During the test time, the models determine whether a test sample is normal or abnormal by detecting outliers from the probability distribution of training data. OCGAN [5] improves robustness using a denoising autoencoder and learns latent space that exclusively represents a given class utilizing two discriminators. Contrary to existing models where a discriminator learns good representations guided by the generator, DGAD [30] makes the generator learn better representations with the help of the discriminator. However, when detecting subtle defects, the anomaly detection algorithms based on the generative models have a limitation. These models learn mainly low-level representations, so generate well-reconstructed features similar to normal samples although the input test sample is abnormal. For practical applications, recent works [31]–[34] that utilize ImageNet pre-trained networks have been studied. These algorithms do not have a training process and just inference to compute the scores. Using pre-trained CNNs with multi-scale feature pyramid pooling for feature extraction, SPADE [31] utilizes k-nearest neighbor for retrieving normal images which are most similar to a test sample and PaDim [33] uses Mahalanobis distance metric with multivariate Gaussian distribution for anomaly detection. The recent success of self-supervised learning has been demonstrated effective in anomaly detection. In self-supervised learning based anomaly detection, the models have been used to learn deep representations by predicting geometric transformation (e.g., rotation, crop, resize) [2], [35], [36] and appearance transformation. In one-class classification settings [37], [15] using contrastive learning contrasts the sample with distributionally-shifted augmentations of itself. Otherwise, CutPaste [38] is to design an augmentation strategy generating local irregular patterns unlike [15] to detect irregular patterns of anomalies. Then [38] trains the model to identify these local irregularities for generalization to unseen real defects at test time. CutPaste generates anomalous patterns by using augmentation such as CutOut [39] or CutPaste which is extracting a rectangular patch from a random area of

an image and pasting it on a random area of the image. Unlike CutPaste, we proposed a general framework that leverages dynamic local augmentations that are automatically determined to generate effective abnormal samples.

B. AUGMENTATION FOR SELF-SUPERVISED LEARNING

Many conventional self-supervised learning methods utilize augmentation to learn the representation. Self-supervised learning approach, e.g., Simsiam [16], BYOL [14], MOCO [12], [13], SimCLR [17], inputs a positive pair of an image by applying a different transformation to one image. Existing works transformed images with Geometric Augmentations [40], Color Augmentation, Blurring Augmentation to create positive pairs. Augmentation methods have been used to generate positive pairs, but a few works [15], [35] used augmentation as negative pair to tackle the image-level Out Of Distribution task by choosing shifting transformation to train the self-supervised learning model.

C. UNCERTAINTY

Our scoring function has been inspired by study of uncertainty [41]–[43]. For practical application on industrial vision inspection and medical diagnosis, uncertainty is a great risk of applying machine learning. Reference [42] has proven that Epistemic uncertainty and Aleatoric uncertainty improve the performance and can measure uncertainty of data and model. Monte Carlo Dropout [44] showed a simple way to measure Epistemic uncertainty at test time.

III. METHOD

In this section, we describe our proposed method in detail. For a given training dataset $\mathcal{D}_{tr} = \{x_m\}_{m=1}^{M}$ that are anomaly-free images, the goal of our framework is to train the model for detecting anomalies in test images $\mathcal{D}_{ts}$. We first introduce our framework based on self-supervised learning. Our model is trained to learn the global and dense representations on the training dataset which consists of the normal data only. Unlike the self-supervised (especially *"contrastive"*) learning methods, we define a negative sample as a sample with dynamic local augmentation to effectively detect anomalies in test images. We also present a novel scoring function to detect the unpredictable abnormal data as well as the predictable abnormal data. In Figure 1, we present the overall architecture. Our framework learns the global and dense representations using self-supervised learning with only positive pairs ($t(x)$ and $t'(x)$) and is also trained to detect anomalies with negative pairs (x and $t_{local}(x)$). To generate the negative sample ($t_{local}(x)$), we apply *dynamic local augmentation* to the sample. To consider a category and a location of an image, various hard local augmentations (e.g., rotation) are adaptively applied. After the training, we measure the regression error score which is the difference between the average feature map of training images and the feature map of a test image. Also, we compute variance per pixel using MC Dropout, and likewise, measure the difference between the average variance map of the training images and the variance map of the test image. The difference maps are computed for each layer (yellow, orange, green squares in the scoring function box), and we finally obtain the anomaly score map for pixel-level anomaly detection.

A. PRELIMINARIES

Before discussing our method, we briefly introduce the self-supervised learning framework which our framework is based on.

1) TERMINOLOGY

Negative pair means the pair between a normal sample and a pseudo abnormal sample generated by dynamic local augmentation. *Negative sample* is a pseudo abnormal sample that is the same as the locally augmented sample. Meanwhile, *positive pair* means the pair between the normal samples during the training. *Hard augmentation*, e.g., rotation, means pushing the sample away from the original by the strong magnitude of the augmentation to generate the negative sample (pseudo abnormal sample). *Predictable anomaly* at test time is an anomaly that has a similar feature representation to a pseudo abnormal sample by dynamic local augmentation. An *unpredictable anomaly* is the opposite of a predictable anomaly which is not predictable by the negative samples.

2) PIPELINE

Given unlabeled data, many approaches for self-supervised representation learning have emerged. MoCo [12], [13] and SimCLR [17] employ contrastive learning where the features of different images (negative pairs) are pulled away while attracting those of the same image's two views (positive pairs). Unlike contrastive learning, BYOL [14] leverages only positive pairs and uses a moving average network to prevent collapsing. Otherwise, SimSiam [16] which is based on simple Siamese networks uses neither negative pairs nor the moving average network. Instead, SimSiam leverages a stop-gradient operation to prevent collapsing solutions. In the anomaly detection task, the training dataset consists of only normal data, and we train/test a model separately for each category. For this reason, the categories of the different images are the same, so the negative pairs are not effective to prevent collapsing solutions, unlike SimCLR. Since the training data, e.g., MVTec AD, is not a large-scale dataset, our framework does not rely on large mini-batch training, unlike BYOL. Consequently, we employ SimSiam architecture as the pipeline of our framework.

3) LOSS FUNCTION

To learn the global feature of an image, our framework first maximizes the cosine similarity of positive pairs like SimSiam. Different from SimSiam, our framework sets two randomly augmented views not only from the same image but also different images as positive pair, since the training data, e.g., MVTec AD, consists of the same category in an anomaly detection task. Second, we observe that learning local features of an image is important in anomaly detection.

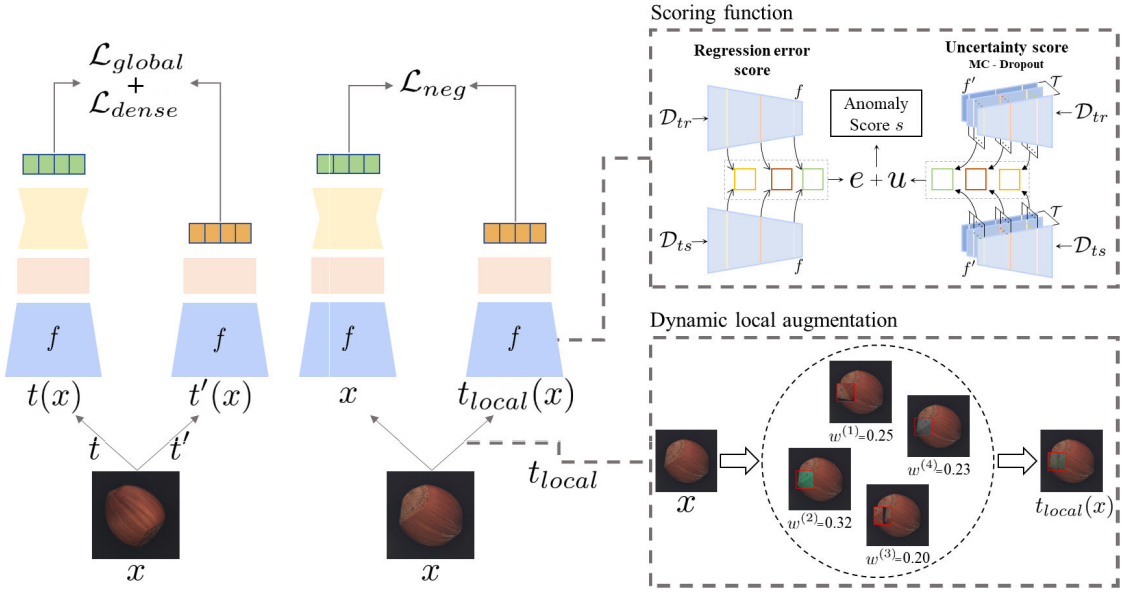

FIGURE 1. Overall architecture. Our framework learns the global and dense representations using self-supervised learning with only positive pairs ($t(x)$ and $t'(x)$) and is also trained to detect anomalies with negative pairs (x and $t_{local}(x)$). To generate abnormal sample, we apply *dynamic local augmentation*. Finally, we measure the regression error score and the uncertainty score using MC Dropout.

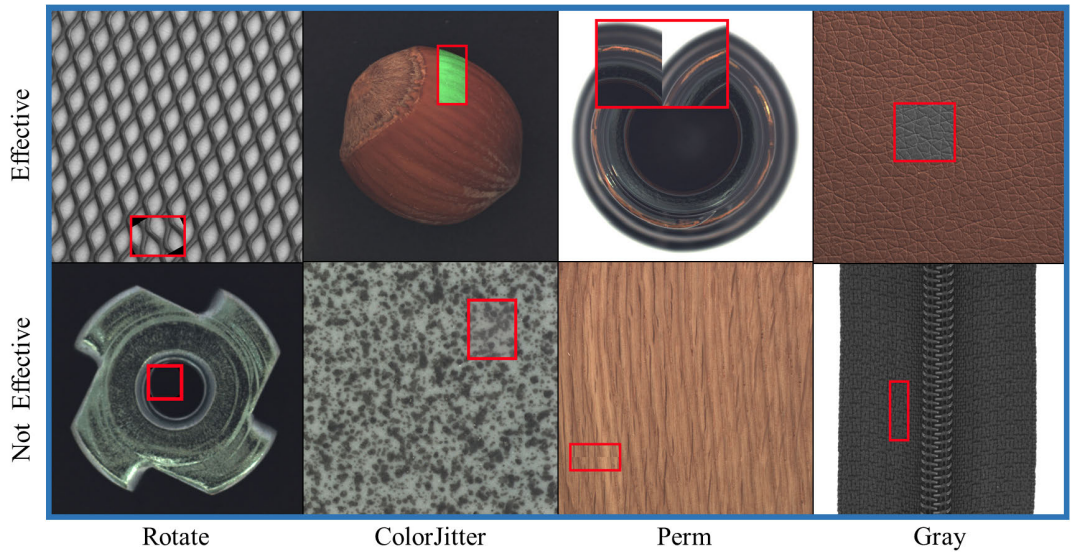

FIGURE 2. Results of *effective* (top row) / *not effective* (bottom row) examples for generating the negative samples. For each type of local augmentation, we observe that the augmented location and the category of the image are important for each local augmentation to generate anomaly effectively.

Recently, dense contrastive learning [11] is presented for self-supervised learning method to directly optimize a pairwise contrastive (dis)similarity loss at the pixel level. We leverage the concept of DenseCL [11] using only positive pairs to effectively detect anomalous regions of an image.

B. SELF-SUPERVISED LEARNING FOR ANOMALY DETECTION

The idea of self-supervised learning is to learn the representations from two randomly augmented views for unlabeled data. To train a model from scratch, our framework optimizes the losses for the global feature and the dense feature using only positive pairs, mentioned in Section III-A.

Given a batch of samples $\mathcal{B} = \{x_k\}_{k=1}^{B}$ and a distribution of image augmentations $\mathcal{T}$, our framework takes two randomly augmented views $v_i \triangleq t(x_i)$ and $v'_j \triangleq t'(x_j)$ as inputs where the image augmentations $t \sim \mathcal{T}$, $t' \sim \mathcal{T}$ and an index of the samples $i, j \in \{1, \ldots, B\}$. Note that the category of the training data is the same in the anomaly detection task, so we set v_i and v'_j as positive pairs although they are transformed from either the same image or different images. We set an encoder network f which is comprised of a backbone and a

projection MLP head and shares the parameters between two views. We denote a prediction MLP head as h and the two output vectors as $z'_j \triangleq f(v'_j)$ and $p_i \triangleq h(f(v_i))$. When calculating the similarity between two outputs, cosine similarity which has been justified to be effective in many self-supervised learning frameworks [12], [13], [16], [17] is used. The negative cosine similarity between them is as follows:

$$D\left(p_i, z'_j\right) = -\frac{p_i}{||p_i||_2}\frac{z'_j}{||z'_j||_2}. \tag{1}$$

Following [16], we symmetrize (1) and define the global self-supervised loss considering stop-gradient (SG) operation to treat z'_j and z_i as a constant during the training as:

$$\mathcal{L}_{global} = \frac{1}{2}D\left(p_i, SG(z'_j)\right) + \frac{1}{2}D\left(SG(z_i), p'_j\right). \tag{2}$$

In anomaly detection, the dense features are also learned to detect anomalous regions at the pixel-level. We employ the loss function for dense features in [11], but our framework uses only positive pairs to learn dense features. Unlike $\mathcal{L}_{global}$ using all pairs of images within mini-batch as input (v_i, v'_j), only two different views of the same image (v_i, v'_i) are used to calculate the dense self-supervised loss $\mathcal{L}_{dense}$. For dense self-supervised loss, we set an encoder network f_d which consists of the same backbone as f and a projection convolution layer head. $h_d(\cdot)$ is denoted as a prediction convolution layer head and D_d is the negative cosine similarity between the corresponding dense feature vectors from the two views following [11]. Since the spatial size of the feature map, e.g., $h_d\left(f_d(v_i)\right)$ and $f_d(v'_i)$, is $S \times S$, we compute the negative cosine similarity matrix $\Delta \in \mathbb{R}^{S^2 \times S^2}$, i.e., $D_d\left(h_d\left(f_d(v_i)\right), f_d(v'_i)\right)$, $D_d\left(f_d(v_i), h_d\left(f_d(v'_i)\right)\right)$. The dense self-supervised loss is as follows:

$$\mathcal{L}_{dense} = \frac{1}{2}D_d\left(h_d\left(f_d(v_i)\right), SG\left(f_d(v'_i)\right)\right) + \frac{1}{2}D_d\left(SG\left(f_d(v_i)\right), h_d\left(f_d(v'_i)\right)\right). \tag{3}$$

C. DYNAMIC LOCAL AUGMENTATION

Next, we present a novel *dynamic local augmentation* to generate negative pairs of the images on the normal training dataset in anomaly detection. The goal of anomaly detection is detecting anomalies in the local regions of an image. Self-supervised learning on only normal training data has limitations, since the model has never been trained with abnormal data. While we train the model to learn the representations of the images by optimizing $\mathcal{L}_{global}$ and $\mathcal{L}_{dense}$, we additionally apply *dynamic local augmentation* to the samples for effectively detecting anomalous regions of the images. Previous works observed that "hard" augmentations, e.g., rotation, is harmful and unused for standard contrastive learning. So we define the hard augmentation at a random location of an image as $\mathcal{T}_{local}$.

1) LOCAL AUGMENTATION

We use various local augmentations, i.e., $\{T^{(k)}_{local}\}^K_{k=1}$, to generate delicate anomalous images. In our method, we use the combination of the following local augmentations to train the model. These augmentations are known for hard augmentation that is useful to generate pseudo abnormal images.

llFirst, we set *Rotation* transformation as one of the hard local augmentations. Rotation transformation is known to be harmful to conventional contrastive learning, because of the hardness of the augmentation. Second, we employ *ColorJitter* transformation to change the color of the local region. We observe that the defects are often revealed as the locally changed color. Third, we consider *Perm* transformation which randomly permutes each part of the evenly partitioned image. Fourth, we add *Grayscale* transformation to easily detect a defect, e.g., hole, crack, and gray stroke, which is usually gray.

Besides these augmentations, we can apply more effective hard augmentation as dynamic local augmentation if the augmentation is hard to transform an image.

2) DEFINITION OF POSITIVE/NEGATIVE PAIR

For our training objective, we use both negative pair and positive pair. Following terminology in Section III-A, the positive/negative pair can be described as the notations: *Positive pair* is the pair between $t(x_i)$ and $t'(x_j)$, where $t \sim \mathcal{T}$, $t' \sim \mathcal{T}$, and $i, j \in \{1, \ldots, B\}$ is an index of the image. Likewise, *negative pair* is the pair between x_i and $t_{local}(x_j)$, where $t_{local}(x_j)$ is the weighted sum of the $t^{(k)}_{local}(x_j)$, $t^{(k)}_{local} \sim \mathcal{T}^{(k)}_{local}$, and k is an index of the hard local augmentation.

Since the hard augmentation is applied at a random location, we observe that some hard local augmentations have no effect at all in certain locations as shown in the bottom row of Figure 2. For example, if rotation is applied at a local region that is without shape (e.g., the center of metal nut and bottle classes, the smooth surface of capsule and leather classes), we observe that there is little change (e.g., the bottom row of the "Rotate" column in Figure 2).

For this reason, we define hard local augmentations as $\{\mathcal{T}^{(1)}_{local}, \ldots, \mathcal{T}^{(K)}_{local}\}$ where K is the number of hard local augmentations, and *adaptively* apply them considering a category and a random location of an image. Our intuition is that the category and the random location of the image affect the effectiveness of the local augmentations to generate the negative sample. To adaptively apply the local augmentations to the image, we propose dynamic local augmentation for a delicate negative sample. We also present the concept of "strong/weak" augmentation for dynamic local augmentation. Strong/weak augmentation is the strength of the augmentation applied to an image. At a certain region of the image, if our framework determines that a local augmentation is useful/useless to generate anomalies, the local augmentation is strongly/weakly applied to the image. To determine whether a local augmentation is strongly or weakly applied, we measure the difference between the original image and the

local augmented image as follows:

$$d^{(k)}(x_i) = MSE\left(t^{(k)}_{local}(x_i), x_i\right), \tag{4}$$

where $t^{(k)}_{local} \sim \mathcal{T}^{(l)}_{local}$, $MSE(\cdot,\cdot)$ is mean squared error, and k is an index of the hard local augmentation. To generate the weights of local augmentations, we define the weights as follows:

$$w^{(k)} = \frac{d^{(k)}(x_i)}{\sum_{k=1}^{K} d^{(k)}(x_i)}, \tag{5}$$

where $w^{(k)} \in [0, 1]$ means the strength for each local augmentation. Next, we define dynamic local augmentation for an input image x_i as follows:

$$t_{local}(x_i) = \sum_{k=1}^{K} w^{(k)} t^{(k)}_{local}(x_i), \tag{6}$$

where $w^{(k)} t^{(k)}_{local}(x_i)$ is element-wise multiplication between locally augmented image and the corresponding weight.

In Figure 3, we show the example of dynamic local augmentation. We denote w as the weight of the local augmentation. Our dynamic local augmentation is the adaptively weighted sum of the local augmentations. It considers the category and the location of the image to generate a more effective negative sample. In the top row of the figure, we observe that the local augmentation is applied to the top location of the bottle image (red line) and "Rotate" and "Perm" are more effective than other augmentations ($w = 0.29$, $w = 0.35$). On the contrary, "ColorJitter" is the most effective augmentation for the center of wood class ($w = 0.36$) as shown in the bottom row of the figure. ColorJitter is twice as strong as the weakest one (Perm) in the example.

We finally present the loss for the negative pair as follows:

$$\mathcal{L}_{neg} = \frac{1}{2} D\left(p_i, SG(z_j)\right) + \frac{1}{2} D\left(SG(z_i), p_j\right), \tag{7}$$

where $z_i \triangleq f(x_i)$, $p_i \triangleq h(z_i)$, $z_j \triangleq f\left(t_{local}(x_j)\right)$, $p_j \triangleq h(z_j)$, and $i, j \in \{1, 2, \ldots, B\}$. Our framework learns detecting the anomalies of the images by dynamic local augmentation as well as the representations of the images by self-supervised loss. Final loss function is as follows:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{global} + \lambda_2 \mathcal{L}_{dense} - \lambda_3 \mathcal{L}_{neg}, \tag{8}$$

where $\lambda_1, \lambda_2, \lambda_3 \in [0, 1]$ are hyperparameters and more details are in Section IV.

D. SCORE FUNCTIONS FOR ANOMALY DETECTION

We define the score functions for detecting anomalies upon the model learned by our proposed training objective. We propose two score functions: 1) regression error between the feature map from training data and the feature map from test data, 2) the predictive uncertainty using MC dropout.

1) REGRESSION ERROR FOR SCORING

In anomaly detection, the training data consists of only normal data, e.g., MVTec AD. So the intuition is that the feature maps of training images from each layer differs from the feature maps of abnormal images on test data. Following [45], the feature maps from each layer have different characteristics, so we compute the regression errors for each layer and average the error values per pixels after upsampling the difference maps for each layer to the same size (224×224). We employ ResNet-18 [46] as backbone network and denote the feature maps from the last layer of each stage as $h^{(l)}$ where $l \in \{1, 2, 3\}$ is an index of the stage. We denote the feature maps from the last layer of each stage on the test dataset as $h^{(l)}_{ts}$. At test time, the regression score is the mean difference between the feature map from the test image and the average feature map from all training images for each layer as follows:

$$e_{(r,c)} = \frac{1}{L} \sum_{l=1}^{L} ||\mu^{(l)}_{(r,c)} - h^{(l)}_{ts,(r,c)}||_2^2, \tag{9}$$

where (r, c) is a pixel of the image, L is the number of stages, $\mu^{(l)}_{(r,c)}$ is the average feature value of all training data at the pixel (r, c), i.e., $\mu^{(l)}_{(r,c)} = \frac{1}{M} \sum_{i=1}^{M} h^{(l)}_{i,(r,c)}$, and M is the number of training data. Note that after computing the differences for each layer, we upsample each difference map to 224×224 and average them per pixel. Also, we do not use the first stage of ResNet-18 because of the resolution size, so three stages (2nd, 3rd, 4th stage) of ResNet-18 are used in (9), i.e., $L = 3$.

2) UNCERTAINTY FOR SCORING

We compute the predictive uncertainty for each pixel using MC Dropout. Upon MC Dropout [44], we apply dropout at test time to compute the model's uncertainty. The intuition is that since the model is trained on only normal data, the model's uncertainty will be high when the input is abnormal test data. We also compute the model's uncertainty for each layer like Regression Error mentioned above. During MC Dropout, T forward passes proceed and we compute the uncertainty at each pixel for each layer as follows:

$$var^{(l)}_{(r,c)} = \frac{1}{T} \sum_{t=1}^{T} \left(\hat{h}^{(l)}_{t,(r,c)}\right)^2 - \left(\frac{1}{T} \sum_{t=1}^{T} \hat{h}^{(l)}_{t,(r,c)}\right)^2, \tag{10}$$

where $\hat{h}^{(l)}_{t,(r,c)}$ is the tth output out of T times at the pixel (r, c) for the lth layer on an image. After upsampling the feature maps consisting of $var^{(l)}_{(r,c)}$ for the lth layer, called variance map, to the resolution (224×224), we compute the difference between the average variance map of the training images and the variance map of the test image for each layer like (9), and we use the difference of uncertainties at each pixel as the score. The equation of the score is as follows:

$$u_{(r,c)} = \frac{1}{L} \sum_{l=1}^{L} ||\overline{var}^{(l)}_{tr,(r,c)} - var^{(l)}_{ts,(r,c)}||_2^2, \tag{11}$$

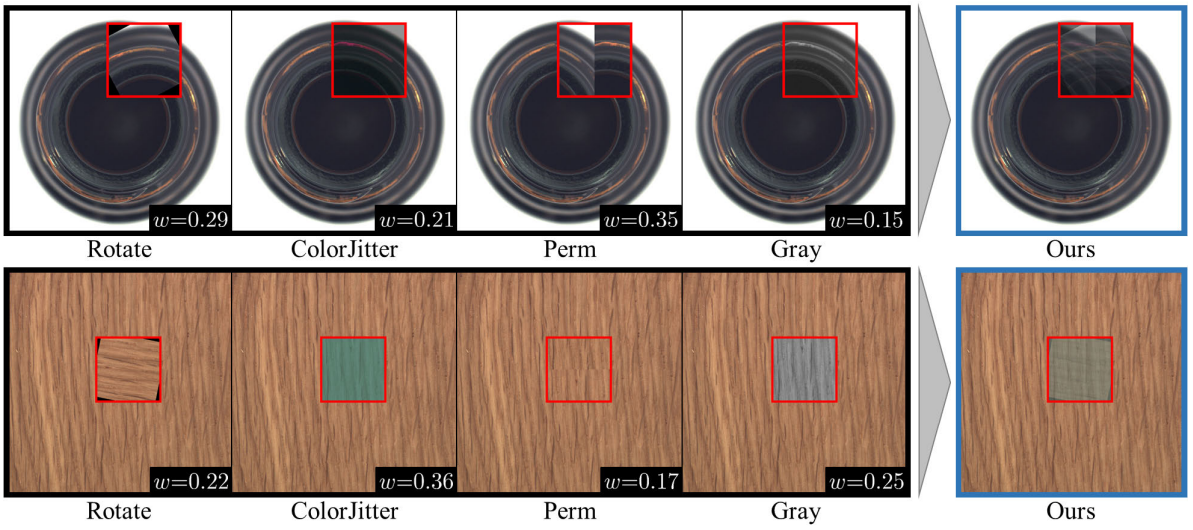

FIGURE 3. Local augmentations (e.g., Rotate, ColorJitter, Perm, Gray) and our dynamic local augmentation.

where $\overline{var}^{(l)}_{tr,(r,c)}$ is the average variance value at pixel (r, c) of all the train images and $var^{(l)}_{ts,(r,c)}$ is a variance value on a test image. Finally, the total scoring function at the pixel (r, c) is as follows:

$$s_{(r,c)} = \alpha_1 e_{(r,c)} + \alpha_2 u_{(r,c)}, \tag{12}$$

where $\alpha_1, \alpha_2 \in [0, 1]$ is the coefficients for each score. Consequently, we use the final score to detect anomalies in the following experiments.

E. IMPLEMENTATION DETAILS

1) UPSAMPLING FEATURE MAPS FOR COMBINING INFORMATION FROM EACH LAYER

For the score of a pixel, we compute the difference of feature values (i.e., regression error) and the difference of variances in the pixel. We upsampled each difference map to 224×224 and average them per pixel after computing the differences for *each layer*. We employ ResNet-18 [46] and use three feature maps from the last layers of each stage. Since we leverage only the feature maps from the last layers of the second, third, and fourth stages, we denote these feature maps as C2, C3, and C4. To average three difference maps from each layer (C2, C3, and C4), we upsample the difference maps generated from C2 (56×56), C3 (28×28), and C4 (14×14) to the same size (224×224).

2) LOSS FOR DENSE REPRESENTATIONS

Our model adopted Dense Contrastive Learning [11] with positive pair of the feature maps from two views of an image. For each view v_i and v'_i where i is an index of the image, the backbone network extracts the feature maps $F_i, F'_i \in R^{S_h \times S_w \times C}$. The feature maps F_i and F'_i are from the last layer of the third stage in ResNet [46], and we defined these feature maps as C3 as mentioned above. From the feature maps, each vector's corresponding index is retrieved:

$$c_i = \underset{s'}{\text{argmax}}\ sim(F_{i,s}, F'_{i,s'}), \tag{13}$$

where $F_{i,s} \in R^{1\times 1\times C}$ is the s^{th} vector of the feature map F_i, and c_i is an index which is the index of the vector corresponding to the vector whose index is s. Dense loss is computed with the corresponding vector from two different output feature maps of the projection layer f_d and prediction layer h_d as mentioned in Section III-B. Dense loss is formulated with corresponding index pair from the backbone feature map:

$$\begin{aligned}\mathcal{L}_{dense} = &\frac{1}{2S_hS_w}\sum_{s=1}^{S_h\times S_w} D_d\left(h_d\left(f_d(v_i)\right)_s, SG\left(f_d(v'_i)_{c_i}\right)\right)\\ &+\frac{1}{2S_hS_w}\sum_{s=1}^{S_h\times S_w} D_d\left(SG\left(f_d(v_i)_{c_i}\right), h_d\left(f_d(v'_i)\right)_s\right),\end{aligned} \tag{14}$$

where the index pair (c_i, s) is derived from (13), and $f_d(v_i)_{c_i}$, $h_d(f_d(v_i))_s$ are the c_i^{th}, s^{th} feature vector of the feature maps $f_d(v_i)$, $h_d(f_d(v_i)) \in R^{S_h\times S_w\times C}$ respectively.

IV. EXPERIMENTS

We demonstrate the effectiveness of our method for detecting anomalies in the images. The experiments are conducted on MVTec Anomaly Detection dataset [47] that is designed for the segmentation of anomalous regions, since it provides pixel-level annotations on test images. The dataset consists of 5 texture and 10 object categories and is composed of only normal images for training and both normal and abnormal images for testing. MVTec AD dataset provides over 5000 high-resolution images where the input images are resized to 256×256 images. We compare our method with various anomaly detection methods (e.g., generative model, teacher-student model) on the dataset.

Setup: In all experiments, we use ResNet-18 [46] as the backbone of our self-supervised learning framework. For a projection and a prediction, we use MLP for global representation and dynamic local augmentation, and convolution layer for dense representation. Our framework is trained from

TABLE 1. AUROC (%) on MVTec anomaly detection dataset [47]. For each category, it measures pixel-wise localization AUC. Our framework that is trained from scratch achieves the best performance on average. We highlight the best performance in boldface.

Category		AE_{SSIM}	AE_{L2}	VAE	AnoGAN	U-Student	Ours
texture	carpet	87	59	59.7	54	86	**89.4**±0.75
	grid	**94**	90	61.2	58	60	88.1±0.95
	leather	78	75	67.1	64	93	**98.5**±0.17
	tile	59	51	51.3	50	87	**91.9**±0.55
	wood	73	73	66.6	62	68	**89.2**±0.17
	average	78.2	69.6	61.18	57.6	78.8	**91.42**
object	bottle	**93**	86	83.1	86	85	91.8±0.17
	cable	82	86	83.1	78	73	**88.3**±1.46
	capsule	94	88	81.7	84	82	**96.5**±0.06
	hazelnut	**97**	95	87.7	87	91	96.2±0.21
	metal nut	89	86	78.7	76	58	**92.6**±0.50
	pill	91	85	81.3	87	90	**96.4**±0.30
	screw	96	96	75.3	80	90	**97.2**±0.10
	toothbrush	92	93	91.9	90	81	**95.8**±0.0
	transistor	**90**	86	75.4	80	85	88.3±1.47
	zipper	88	77	71.6	78	90	**95.4**±0.23
	average	91.2	87.8	80.98	82.6	82.5	**93.85**
average		86.87	81.73	74.38	74.27	81.27	**93.04**

scratch on MVTec AD dataset in the experiments. We set that batch size is 200, the optimizer is SGD with momentum 0.9, weight decay is 5×10^{-4}, the number of training epochs is 200, a base learning rate *lr* is 0.02. The size of the random local region for dynamic local augmentation is ranged from 50 to 128, and we can detect anomalies at various scales. Also, the height and the width of the region for the local augmentation can be different within the range. The coefficients λ_1, λ_2, and λ_3 in (8) are respectively 1, 0.1, 0.5 in our experiments. For scoring in (12), we set $\alpha_1 = 1$ and $\alpha_2 = 0.1$. When choosing the hyper-parameters, we fixed the learning rate, weight decay, and epochs which are the same as the backbone model [16]. The hyper-parameters that we chose are λ_1, λ_2, and λ_3 in (8) and α_1 and α_2 in (12). In anomaly detection, the training set consists of only normal samples, so the hyper-parameter tuning using the validation set is difficult in the anomaly detection task. In contrast to other baselines, our framework can generate the pseudo abnormal samples using local augmentations, so we did hyper-parameter tuning on the validation set using normal samples and pseudo abnormal samples. In (8), we fixed λ_1 as 1 and tuned λ_2, λ_3 in the range of 0.1∼0.5. In (12), we tuned α_1, α_2 in the range of 0.1∼1.0 and chose the final coefficient value using the validation set.

A. MAIN RESULTS

In Table 1, we evaluate the anomaly detection performance on MVTec Anomaly Detection (MVTec AD) dataset. We report the average AUROC (%) of three independent runs with different random seeds in Table 1. The average AUROCs for texture, object, and all categories are reported.

We compare our proposed method with the baselines presented by [47]. These baselines include AE_{SSIM} [27], AE_{L2} [27], VAE [48], AnoGAN [3], and we also compare with U-Student [49] that is based on student-teacher framework.

In Table 1, we show that our method achieves the best performance for most categories as well as the average of all categories. Our method outperforms the second-highest baseline by 12.66% in texture category, by 2.53% in object category. Our framework especially outperforms the second-highest method for wood in texture category by 16.4% and for zipper in object category by 7.5%. The average AUROCs of the baselines in wood and zipper classes are 68.52% and 80.92% respectively, however the performance of our method is 89.4% and 95.5% respectively. Overall, our method consistently achieves high AUROC (%) for all the categories.

B. DISCUSSION

In this section, we conducted ablation study to discuss the contribution of our framework. We first provide the experimental result to validate the effectiveness of dynamic local augmentation and then analyze each component of our method in Table 2.

Our framework is the self-supervised learning framework with dynamic local augmentation. In Table 2, we observed

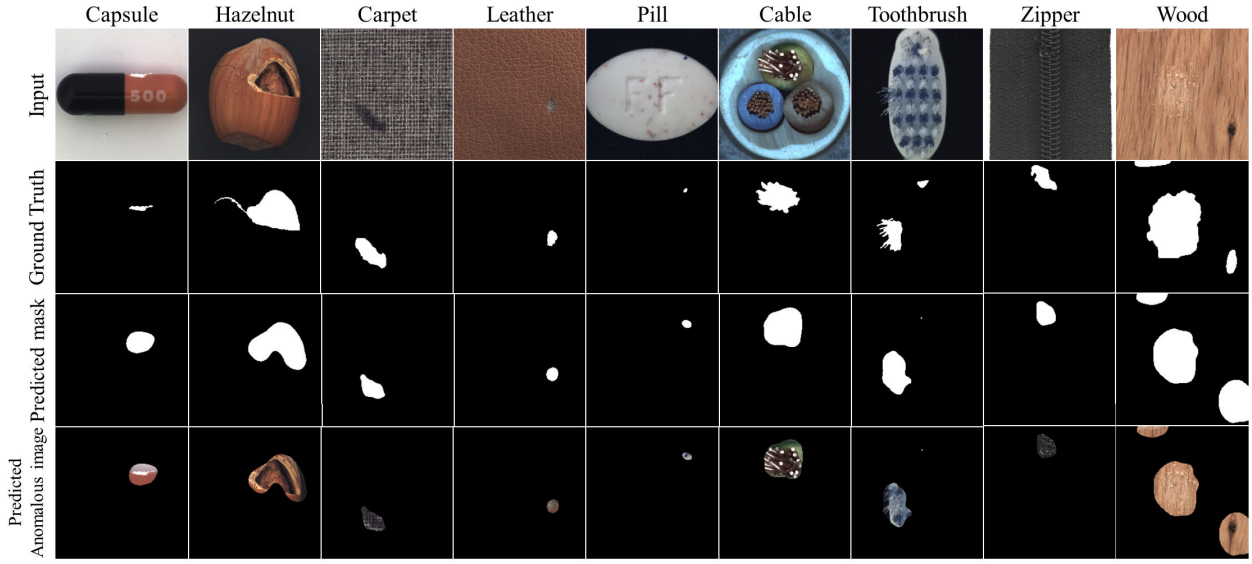

FIGURE 4. Results of defect localization on capsule, hazelnut, carpet, leather, pill, cable, and toothbrush classes. From the first row to the last row, input images, ground truth mask in white, predicted mask in white, and predicted anomalous images.

TABLE 2. AUROC (%) for ablation study on MVTec AD dataset [47]. The first∼second row shows the performance using only self-supervised learning without negative pairs. We denote using fixed local augmentation (not dynamic) as *LA* and using dynamic local augmentation as *DLA*. Last row is our method and shows the best performance on pixel-wise localization.

Components				Category		
$\mathcal{L}_{global}$	$\mathcal{L}_{dense}$	$\mathcal{L}_{neg}$		texture	object	Total avg.
		LA	*DLA*			
✓	-	-	-	85.64	84.13	84.63
✓	✓	-	-	86.86	86.90	86.89
✓	-	-	✓	88.64	89.60	89.28
✓	✓	✓	-	77.40	85.68	82.92
✓	✓	-	✓	**91.42**	**93.85**	**93.04**

that the model trained with self-supervised learning using only global and dense loss (the first∼second row in Table 2) tends to have poor detecting anomalies at test time. Since the model has never learned to detect anomalies on the training dataset, self-supervised learning without dynamic local augmentation has difficulty detecting the abnormal patterns. We denote a fixed local augmentation (not dynamic) and a dynamic local augmentation as *LA* and *DLA* respectively. The fourth row and the last row (Ours) in Table 2 show that the model with *DLA* (the last row) performs an average of 9.90% higher than with *LA*. We observe that the model with *LA* underperforms the other models, and it shows that the local augmentation is effective as a negative sample when dynamically applied. In the third row and the last row of Table 2, we observe that dense loss $\mathcal{L}_{dense}$ is an important component, especially in the object category.

Our intuition is that a model achieved competitive performance when using a negative sample generated by a *proper* hard local augmentation that is neither too weak nor too strong at a random location. If we use the negative sample with weak augmentation, e.g., "not effective" in Figure 2, our framework cannot learn the representation of an anomaly. On the contrary, if we leverage the negative sample with too strong augmentation, e.g., $w^{(k)} = 1$, where $k \in \{1, 2, 3, 4\}$ in Equation 5, our framework is difficult to detect the abnormal region which includes *imperceptible* anomaly, although the ground truth of the region is abnormal at test time. We define "too strong augmentation" as applying all hard local augmentations (e.g., Rotate, ColorJitter, Perm, and Gray) to a sample without considering the weight of each augmentation (i.e., the weight $w^{(k)} = 1$). In Table 2, we observed that our framework with dynamic local augmentation (the last row in Table 2) achieved the best performance. If we train the model using the negative sample with static (fixed) local augmentation (*LA*), i.e., too strong augmentation, the model achieved poor performance as seen in the second row of Table 2. Furthermore, training without local augmentation is more effective than training using static local augmentation (*LA*) as shown in the second and forth row in Table 2. The main intuition from Table 2 is that self-supervised learning method is sensitive to abnormal samples. When static augmentation is randomly applied as shown in Figure 2 of the main paper, not-effective pseudo-abnormal samples can be generated causing a huge drop of performance from 93.04 to 82.92.

In Table 3, we showed the performance of all categories comparing dynamic local augmentation (*DLA*) with static local augmentation (*LA*). Overall, our framework (*DLA*) outperforms *LA*, and the performance gap between them is especially large on some categories such as grid, tile, bottle, and metal nut. The average performance gap on these categories is 25.4%, and the performance gap between our framework and *LA* on texture category (14.02%) is averagely larger than the gap on object category (8.17%). The ablation study on

TABLE 3. The comparison of all categories between *LA* (static local augmentation) and *DLA* (dynamic local augmentation) on MVTec AD dataset [47].

Category		*LA* (not dynamic)	*DLA* (Ours)
texture	carpet	83.2	**89.4**
	grid	64.8	**88.1**
	leather	97.2	**98.5**
	tile	54.6	**91.9**
	wood	87.2	**89.2**
	average	77.40	**91.42**
object	bottle	67.0	**91.8**
	cable	77.9	**88.3**
	capsule	89.8	**96.5**
	hazelnut	93.2	**96.2**
	metal nut	76.4	**92.6**
	pill	92.2	**96.4**
	screw	95.2	**97.2**
	toothbrush	**95.8**	**95.8**
	transistor	75.6	**88.3**
	zipper	93.8	**95.4**
	average	85.68	**93.85**
average		82.92	**93.04**

dynamic local augmentation shows that to learn tight decision boundaries of normal pixels, the effective pseudo abnormal sample needs to be generated. The performance drop on the texture category is bigger than the object class, since static local augmentation is not effective in the texture category causing the wrong decision boundary. Figure 4 shows the result images for some categories.

V. CONCLUSION

We proposed a simple yet effective framework by self-supervised learning with dynamic local augmentation in anomaly detection. For unsupervised anomaly detection, our method learns the global and dense representations using only normal image data. We present a novel augmentation method called dynamic local augmentation to generate pseudo abnormal images and effectively detect the anomalies at test time. Dynamic local augmentation applies suitable augmentations to the local region of the image, considering the category and the location of the image. We also propose two scoring functions using regression error and variance map.

Our method achieved competitive performance for pixel-wise anomaly segmentation. We experiment our model with pixel level anomaly dataset MVTec AD from scratch. In Table 2, we show that a variety of combinations of four losses affect the performance and dynamic local augmentation is helpful but conventional local augmentation interferes with the performance. In Table 3, we show the effectiveness of pseudo abnormal sample by each category. And we observe the performance with comparison between *LA* (not dynamic) and *DLA* (Ours) by each category.

We have studied generating an abnormal sample by dynamic local augmentation. From the experiment results, we have shown the importance of an effective anomaly sample and dense features. Also, we have first explored uncertainty scores for pixel-level anomaly detection with self-supervised learning. For our future works, we will extend our work to wide-scene anomaly detection where many objects appear in one image or video where uncertainty is high. In future works, we will study how to generate the abnormal sample. The shape and the size of dynamic local augmentation will affect the quality of the abnormal samples, and the variety of abnormal patterns are important to make the negative pairs. The proposed method can guide interesting future directions for anomaly detection.

ACKNOWLEDGMENT

(Seungdong Yoa and Seungjun Lee contributed equally to this work.)

REFERENCES

[1] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4393–4402.
[2] L. Bergman and Y. Hoshen, "Classification-based anomaly detection for general data," in *Proc. Int. Conf. Learn. Represent.*, 2019.
[3] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *Proc. Int. Conf. Inf. Process. Med. Imag.* Springer, 2017, pp. 146–157.
[4] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "GANomaly: Semi-supervised anomaly detection via adversarial training," in *Proc. Asian Conf. Comput. Vis.* Springer, 2018, pp. 622–637.
[5] P. Perera, R. Nallapati, and B. Xiang, "OCGAN: One-class novelty detection using GANs with constrained latent representations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2898–2906.
[6] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding Gaussian mixture model for unsupervised anomaly detection," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 2–20.
[7] S. Pidhorskyi, R. Almohsen, and G. Doretto, "Generative probabilistic novelty detection with adversarial autoencoders," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 6822–6833.
[8] D. Abati, A. Porrello, S. Calderara, and R. Cucchiara, "Latent space autoregression for novelty detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 481–490.
[9] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. A. DePristo, J. V. Dillon, and B. Lakshminarayanan, "Likelihood ratios for out-of-distribution detection," 2019, *arXiv:1906.02845*.
[10] E. Nalisnick, A. Matsukawa, Y. W. Teh, and B. Lakshminarayanan, "Detecting out-of-distribution inputs to deep generative models using a test for typicality," vol. 5, p. 5, 2019, *arXiv:1906.02994*.
[11] X. Wang, R. Zhang, C. Shen, T. Kong, and L. Li, "Dense contrastive learning for self-supervised visual pre-training," 2020, *arXiv:2011.09157*.
[12] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," 2020, *arXiv:2003.04297*.
[13] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9729–9738.
[14] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko, "Bootstrap your own latent: A new approach to self-supervised learning," 2020, *arXiv:2006.07733*.
[15] J. Tack, S. Mo, J. Jeong, and J. Shin, "CSI: Novelty detection via contrastive learning on distributionally shifted instances," 2020, *arXiv:2007.08176*.
[16] X. Chen and K. He, "Exploring simple Siamese representation learning," 2020, *arXiv:2011.10566*.
[17] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.

[18] H. Chung, J. Park, J. Keum, H. Ki, and S. Kang, "Unsupervised anomaly detection using style distillation," *IEEE Access*, vol. 8, pp. 221494–221502, 2020.

[19] Y.-G. Kim and T.-H. Park, "Anomaly detection using autoencoder with feature vector frequency map," *IEEE Access*, vol. 9, pp. 73808–73817, 2021.

[20] W. Wang, W. Song, Z. Li, B. Zhao, and B. Zhao, "A novel filter-based anomaly detection framework for hyperspectral imagery," *IEEE Access*, vol. 9, pp. 124033–124043, 2021.

[21] N. Shvetsova, B. Bakker, I. Fedulova, H. Schulz, and D. V. Dylov, "Anomaly detection in medical imaging with deep perceptual autoencoders," *IEEE Access*, vol. 9, pp. 118571–118583, 2021.

[22] M. Sabokrou, M. Fathy, G. Zhao, and E. Adeli, "Deep end-to-end one-class classifier," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 2, pp. 675–684, Feb. 2021.

[23] L. Ruff, J. Kauffmann, R. A. Vandermeulen, G. Montavon, W. Samek, M. Kloft, T. G. Dietterich, and K.-R. Müller, "A unifying review of deep and shallow anomaly detection," *Proc. IEEE*, vol. 109, no. 5, pp. 756–795, May 2021.

[24] Y. Fei, C. Huang, C. Jinkun, M. Li, Y. Zhang, and C. Lu, "Attribute restoration framework for anomaly detection," *IEEE Trans. Multimedia*, early access, Dec. 30, 2021, doi: 10.1109/TMM.2020.3046884.

[25] X. Fang, W. Guo, Q. Li, J. Zhu, Z. Chen, J. Yu, B. Zhou, and H. Yang, "Sewer pipeline fault identification using anomaly detection algorithms on video sequences," *IEEE Access*, vol. 8, pp. 39574–39586, 2020.

[26] Y. Zhang, B. Du, L. Zhang, and S. Wang, "A low-rank and sparse matrix decomposition-based Mahalanobis distance method for hyperspectral anomaly detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 3, pp. 1376–1389, Mar. 2016.

[27] P. Bergmann, S. Löwe, M. Fauser, D. Sattlegger, and C. Steger, "Improving unsupervised defect segmentation by applying structural similarity to autoencoders," 2018, *arXiv:1807.02011*.

[28] P. C. Ngo, A. A. Winarto, C. K. L. Kou, S. Park, F. Akram, and H. K. Lee, "Fence GAN: Towards better anomaly detection," in *Proc. IEEE 31st Int. Conf. Tools With Artif. Intell. (ICTAI)*, Nov. 2019, pp. 141–148.

[29] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "Skip-GANomaly: Skip connected and adversarially trained encoder-decoder anomaly detection," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–8.

[30] X. Xia, X. Pan, X. He, J. Zhang, N. Ding, and L. Ma, "Discriminative-generative representation learning for one-class anomaly detection," 2021, *arXiv:2107.12753*.

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

[32] O. Rippel, P. Mertens, and D. Merhof, "Modeling the distribution of normal data in pre-trained deep features for anomaly detection," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 6726–6733.

[33] T. Defard, A. Setkov, A. Loesch, and R. Audigier, "PaDIM: A patch distribution modeling framework for anomaly detection and localization," in *Proc. ICPR Workshops*, 2020, pp. 475–489.

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

[35] I. Golan and R. El-Yaniv, "Deep anomaly detection using geometric transformations," in *Proc. NeurIPS*, 2018, pp. 9781–9791.

[36] D. Hendrycks, M. Mazeika, S. Kadavath, and D. X. Song, "Using self-supervised learning can improve model robustness and uncertainty," in *Proc. NeurIPS*, 2019, pp. 15663–15674.

[37] P. Perera and V. M. Patel, "Learning deep features for one-class classification," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5450–5463, Nov. 2019.

[38] C.-L. Li, K. Sohn, J. Yoon, and T. Pfister, "CutPaste: Self-supervised learning for anomaly detection and localization," 2021, *arXiv:2104.04015*.

[39] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," 2017, *arXiv:1708.04552*.

[40] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3733–3742.

[41] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proc. NIPS*, 2017, pp. 1–12.

[42] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision," in *Proc. NIPS*, 2017, pp. 1–11.

[43] Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. V. Dillon, B. Lakshminarayanan, and J. Snoek, "Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift," 2019, *arXiv:1906.02530*.

[44] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proc. Int. Conf. Mach. Learn.* PMLR, 2016, pp. 1050–1059.

[45] C. Zhang, S. Bengio, and Y. Singer, "Are all layers created equal?" 2019, *arXiv:1902.01996*.

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

[47] 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. 9592–9600.

[48] C. Baur, B. Wiestler, S. Albarqouni, and N. Navab, "Deep autoencoding models for unsupervised anomaly segmentation in brain mr images," in *Proc. Int. MICCAI Brainlesion Workshop*. Springer, 2018, pp. 161–169.

[49] 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. 4183–4192.

SEUNGDONG YOA received the B.S. degree in computer science from Korea University, in 2017, where he is currently pursuing the master's degree in computer science, studying under Hyunwoo J. Kim. His current research interests include image classification, meta-learning, anomaly detection, self-supervised learning, and computer vision.

SEUNGJUN LEE received the B.S. degree in industrial and management engineering from Korea University, in 2019, where he is currently pursuing the master's degree in computer science, studying under Hyunwoo J. Kim with the Laboratory of Machine Learning and Computer Vision. His current research interests include anomaly detection, visual relationship, and computer vision in drones.

CHIYOON KIM received the B.S. degree in digital information engineering from the Hankuk University of Foreign Studies, in 2021. He is currently pursuing the M.S. degree in computer science with Korea University, where he is studying under Hyunwoo J. Kim. His current research interests include anomaly detection, self-supervised learning, and computer vision.

HYUNWOO J. KIM received the Ph.D. degree in computer science and minor is statistics from the University of Wisconsin–Madison, in 2017. He worked at Amazon Lab126, Sunnyvale, CA, USA. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Korea University. His research interests include geometric deep learning for graphs and manifolds, manifold statistics, machine learning, computer vision, and medical imaging.