Patch SVDD: Patch-level SVDD
for Anomaly Detection and Segmentation

Jihun Yi[0000−0001−5762−6643] and Sungroh Yoon⋆[0000−0002−2367−197X]

Data Science and Artificial Intelligence Laboratory
Electrical and Computer Engineering
Seoul National University, Seoul, South Korea
{t080205,sryoon}@snu.ac.kr

Abstract. In this paper, we address the problem of image anomaly detection and segmentation. Anomaly detection involves making a binary decision as to whether an input image contains an anomaly, and anomaly segmentation aims to locate the anomaly on the pixel level. Support vector data description (SVDD) is a long-standing algorithm used for an anomaly detection, and we extend its deep learning variant to the patch-based method using self-supervised learning. This extension enables anomaly segmentation and improves detection performance. As a result, anomaly detection and segmentation performances measured in AUROC on MVTec AD dataset increased by 9.8% and 7.0%, respectively, compared to the previous state-of-the-art methods. Our results indicate the efficacy of the proposed method and its potential for industrial application. Detailed analysis of the proposed method offers insights regarding its behavior, and the code is available online[1].

1   Introduction

Anomaly detection is a binary classification problem to determine whether an input contains an anomaly. Detecting anomalies is a critical and long-standing problem faced by the manufacturing industry. Anomaly detection is usually formulated as a one-class classification because abnormal examples are either inaccessible or insufficient to model distribution during the training. When concentrating on image data, detected anomalies can also be localized, and anomaly segmentation problem is to localize the anomalies at the pixel level. In this study, we tackle the problems of image anomaly detection and segmentation.

One-class support vector machine (OC-SVM) [1] and support vector data description (SVDD) [2] are classic algorithms used for one-class classification. Given a kernel function, OC-SVM seeks a max-margin hyperplane from the origin in the kernel space. Likewise, SVDD searches for a data-enclosing hypersphere in the kernel space. These approaches are closely related, and Vert et al. [3] showed their equivalence in the case of a Gaussian kernel. Ruff et al. [4] proposed a deep

---

⋆ Correspondence to: Sungroh Yoon (sryoon@snu.ac.kr).
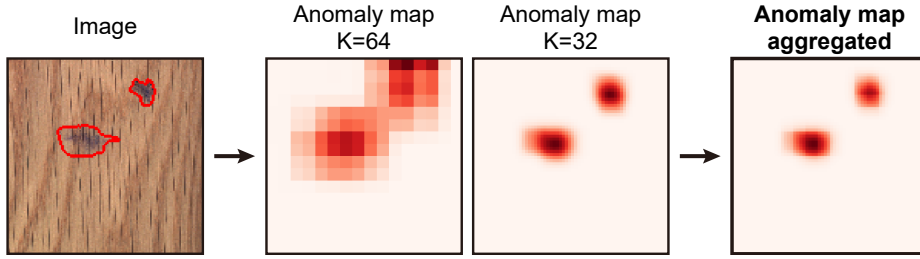[1] https://github.com/nuclearboy95/Anomaly-Detection-PatchSVDD-PyTorch

Fig. 1: **Proposed method localizes defects in an MVTec AD [8] image.** Patch SVDD performs multi-scale inspection and combines the results. As a result, the anomaly map pinpoints the defects (contoured with a red line).

learning variant of SVDD, Deep SVDD, by deploying a deep neural network in the place of the kernel function. The neural network was trained to extract a data-dependent representation, removing the need to choose an appropriate kernel function by hand. Furthermore, Ruff et al. [5] re-interpreted Deep SVDD in an information-theoretic perspective and applied to semi-supervised scenarios.

In this paper, we extend Deep SVDD to a patch-wise detection method, thereby proposing Patch SVDD. This extension is rendered nontrivial by the relatively high level of intra-class variation of the patches and is facilitated by self-supervised learning. The proposed method enables anomaly segmentation and improves anomaly detection performance. Fig. 1 shows an example of the localized anomalies using the proposed method. In addition, the results in previous studies [6,7] show that the features of a randomly initialized encoder might be used to distinguish anomalies. We detail the more in-depth behavior of random encoders and investigate the source of separability in the random features.

2   Background

2.1   Anomaly detection and segmentation

Problem formulation Anomaly detection is a problem to make a binary decision whether an input is an anomaly or not. The definition of *anomaly* ranges from a tiny defect to an out-of-distribution image. We focus here on detecting a defect in an image. A typical detection method involves training a scoring function, $\mathcal{A}_\theta$, which measures the abnormality of an input. At test time, inputs with high $\mathcal{A}_\theta(\mathbf{x})$ values are deemed to be an anomaly. A *de facto* standard metric for evaluating the scoring function is the area under the receiver operating characteristic curve (AUROC), as expressed in Eq. 1 [9].

$$\mathrm{AUROC}\left[\mathcal{A}_\theta\right] = \mathbb{P}\left[\mathcal{A}_\theta(\mathbf{X}_{\mathrm{normal}}) < \mathcal{A}_\theta(\mathbf{X}_{\mathrm{abnormal}})\right]. \tag{1}$$

A good scoring function is, thus, one that assigns a low anomaly score to normal data and a high anomaly score to abnormal data. Anomaly segmentation problem
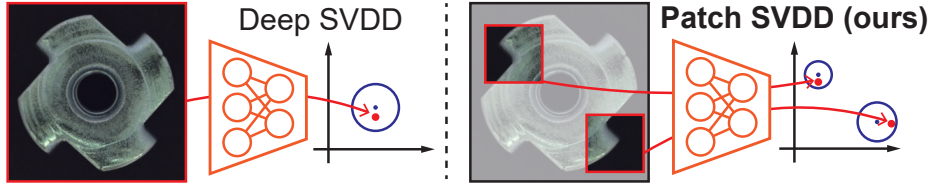
Fig. 2: **Comparison of Deep SVDD [4] and the proposed method.** Patch SVDD performs inspection on every patch to localize a defect. In addition, the self-supervised learning allows the features to form multi-modal clusters, thereby enhancing anomaly detection capability. The image is from MVTec AD [8] dataset.

is similarly formulated; anomaly score is estimated for each pixel (i.e., an anomaly map), and AUROC is measured using the pixels.

Auto encoder-based methods Early deep learning approaches to anomaly detection used auto encoders [10,11,12]. These auto encoders were trained with the normal training data and did not provide accurate reconstruction of abnormal images. Therefore, the difference between the reconstruction and the input indicated abnormality. Further variants have been proposed to utilize structural similarity indices [13], adversarial training [11], negative mining [12], and iterative projection [14]. Certain previous works utilized the learned latent feature of the auto encoder for anomaly detection. Akcay et al. [15] defined the reconstruction loss of the latent feature as an anomaly score, and Yarlagadda et al. [16] trained OC-SVM [1] using the latent features. More recently, several methods have made use of factors other than reconstruction loss, such as restoration loss [17] and an attention map [18].

Classifier-based methods After the work of Golan et al. [19], discriminative approaches have been proposed for anomaly detection. These methods exploit an observation that classifiers lose their confidence [20] for the abnormal input images. Given an unlabeled dataset, a classifier is trained to predict synthetic labels. For example, Golan et al. [19] randomly flip, rotate, and translate an image, and the classifier is trained to predict the particular type of transformation performed. If the classifier does not provide a confident and correct prediction, the input image is deemed to be abnormal. Wang et al. [21] proved that such an approach could be extended to an unsupervised scenario, where the training data also contains a few anomalies. Bergman et al. [22] adopted an open-set classification method and generalized the method to include non-image data.

SVDD-based methods SVDD [2] is a classic one-class classification algorithm. It maps all the normal training data into a predefined kernel space and seeks the smallest hypersphere that encloses the data in the kernel space. The anomalies are expected to be located outside the learned hypersphere. As a kernel function
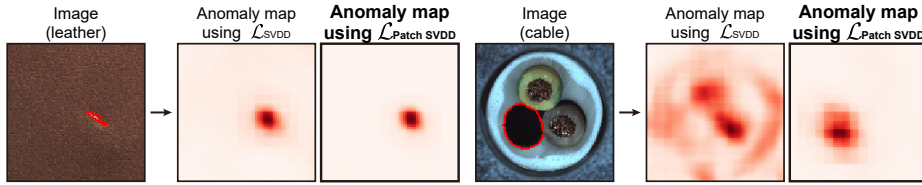
Fig. 3: **Comparison of anomaly maps generated using two different losses.** For a relatively simple image (leather), the encoders trained with either $\mathcal{L}_{\mathrm{SVDD}}$ or $\mathcal{L}_{\mathrm{Patch\ SVDD}}$ both localize the defect (contoured with a red line) well. By contrast, when the image has high complexity (cable), $\mathcal{L}_{\mathrm{SVDD}}$ fails to localize the defect. The image is from MVTec AD [8] dataset.

determines the kernel space, the training procedure is merely deciding the radius and center of the hypersphere.

Ruff et al. [4] improved this approach using a deep neural network. They adopted the neural network in place of the kernel function and trained it along with the radius of the hypersphere. This modification allows the encoder to learn a data-dependent transformation, thus enhancing detection performance on high-dimensional and structured data. Ruff et al. [5] further applied this method to a semi-supervised scenario.

2.2   Self-supervised representation learning

Learning a representation of an image is a core problem of computer vision. A series of methods have been proposed to learn a representation of an image without annotation. One branch of research suggests training the encoder by learning with a *pretext task*, which is a self-labeled task to provide synthetic learning signals. When a network is trained to solve the pretext task well, the network is expected to be able to extract useful features. The pretext tasks include predicting relative patch location [6], solving a jigsaw puzzle [23], colorizing images [24], counting objects [25], and predicting rotations [26].

3   Methods

3.1   Patch-wise Deep SVDD

Deep SVDD [4] trains an encoder that maps the entire training data to features lying within a small hypersphere in the feature space. The authors trained the encoder, f_θ, to minimize the Euclidean distances between the features and the center of the hypersphere using the following loss function:

$$\mathcal{L}_{\mathrm{SVDD}} = \sum_i^N \left\| f_\theta(\mathbf{x}_i) - \mathbf{c} \right\|_2 , \tag{2}$$
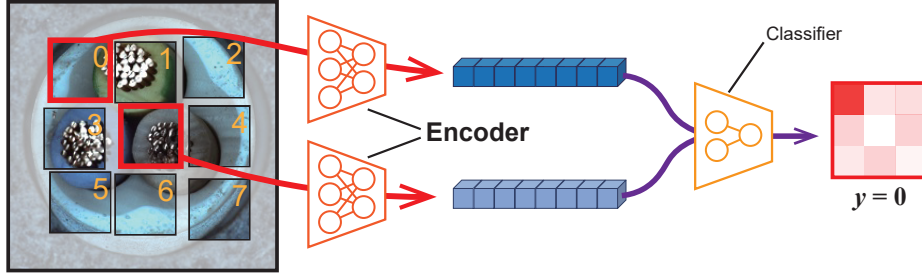
Fig. 4: **Self-supervised learning [6].** The encoder is trained to extract informative features so that the following classifier can correctly predict the relative positions of the patches. Once the training is complete, the classifier is discarded. Note that the two encoders share their weights, as in Siamese Network [27]. The image is from MVTec AD [8] dataset.

where $\mathbf{x}$ is an input image, i is a data index, and N denotes the number of the training data. At test time, the distance between the representation of the input and the center is used as an anomaly score. The center $\mathbf{c}$ is calculated in advance of the training, as shown in Eq. 3. Therefore, the training pushes the features around a single center. Note that the authors removed the bias terms in the network to avoid a trivial solution (i.e., the encoder outputs a constant).

$$\mathbf{c} \doteq \frac{1}{N} \sum_{i}^{N} f_{\theta}^{\text{untrained}}(\mathbf{x}_i). \tag{3}$$

In this study, we extend this approach to patches; the encoder encodes each patch, not the entire image, as illustrated in Fig. 2. Accordingly, inspection is performed for each patch. Patch-wise inspection has several advantages. First, the inspection result is available at each position, and hence we can localize the positions of defects. Second, such fine-grained examination improves overall detection performance.

A direct extension of Deep SVDD [4] to a patch-wise inspection is straightforward. A patch encoder, f_{θ}, is trained using $\mathcal{L}_{\text{SVDD}}$ with $\mathbf{x}$ replaced with a patch, $\mathbf{p}$. The anomaly score is defined accordingly, and the examples of the resulting anomaly maps are provided in Fig. 3. Unfortunately, the detection performance is poor for the images with high complexity. This is because patches have high intra-class variation; some patches correspond to the background, while the others may contain the object. As a result, mapping all the features of dissimilar patches to a single center and imposing a uni-modal cluster weaken the connection between the representation and the content. Therefore, using a single center $\mathbf{c}$ is inappropriate, yet deciding on the appropriate number of multiple centers and the allocation of patches to each center are cumbersome.

To bypass the above issues, we do not explicitly define the centers and allocate the patches. Instead, we train the encoder to gather semantically similar patches by itself. The semantically similar patches are obtained by sampling spatially
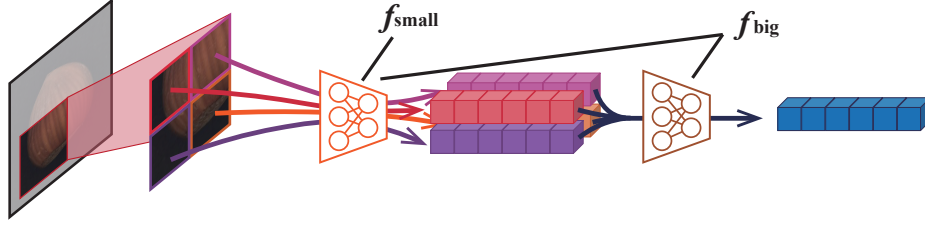
Fig. 5: **Hierarchical encoding.** An input patch is divided into a 2×2 grid of sub-patches, and the sub-patches are independently encoded using the smaller encoder (f_{small}). The output features are aggregated to produce a single feature. The image is from MVTec AD [8] dataset.

adjacent patches, and the encoder is trained to minimize the distances between their features using the following loss function:

$$\mathcal{L}_{\text{SVDD'}} = \sum_{i,i'} \| f_\theta(\mathbf{p}_i) - f_\theta(\mathbf{p}_{i'}) \|_2 , \tag{4}$$

where $\mathbf{p}_{i'}$ is a spatially jittered $\mathbf{p}_i$. Furthermore, to enforce the representation to capture the semantics of the patch, we add the following self-supervised learning.

3.2   Self-supervised learning

Doersch et al. [6] trained an encoder and classifier pair to predict the relative positions of two patches, as depicted in Fig. 4. A well-performing pair implies that the trained encoder extracts useful features for location prediction. Aside from this particular task, previous research [23,26,28] reported that the self-supervised encoder functions as a powerful feature extractor for downstream tasks.

For a randomly sampled patch $\mathbf{p}_1$, Doersch et al. [6] sampled another patch $\mathbf{p}_2$ from one of its eight neighborhoods in a 3×3 grid. If we let the true relative position be $y \in \{0, ..., 7\}$, the classifier C_ϕ is trained to predict the position correctly; i.e., $y = C_\phi(f_\theta(\mathbf{p}_1), f_\theta(\mathbf{p}_2))$. The size of the patch is the same as the receptive field of the encoder. To prevent the classifier from exploiting shortcuts [7] (e.g., color aberration), we randomly perturb the RGB channels of the patches. Following the approach by Doersch et al. [6], we add a self-supervised learning signal by adding the following loss term:

$$\mathcal{L}_{\text{SSL}} = \texttt{Cross-entropy}\left(y, C_\phi\left(f_\theta(\mathbf{p}_1), f_\theta(\mathbf{p}_2)\right)\right). \tag{5}$$

As a result, the encoder is trained using a combination of two losses with the scaling hyperparameter λ, as presented in Eq. 6. This optimization is performed using stochastic gradient descent and Adam optimizer [29].

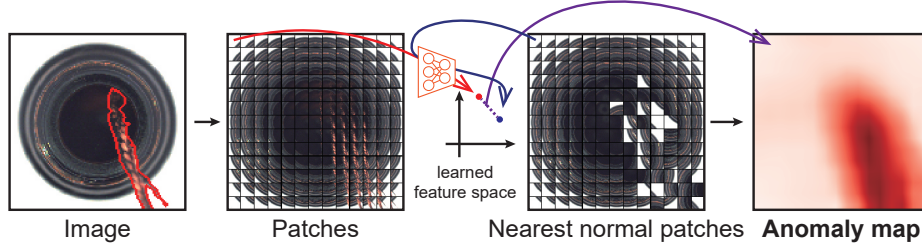$$\mathcal{L}_{\text{Patch SVDD}} = \lambda \mathcal{L}_{\text{SVDD'}} + \mathcal{L}_{\text{SSL}}. \tag{6}$$

Fig. 6: **Overall flow of the proposed method.** For a given test image, Patch SVDD divides the image into patches of size K with strides S and extracts their features using the trained encoder. The L2 distance to the nearest normal patch in the feature space becomes the anomaly score of each patch. The resulting anomaly map localizes the defects (contoured with a red line). The image is from MVTec AD [8] dataset.

3.3   Hierarchical encoding

As anomalies vary in size, deploying multiple encoders with various receptive fields helps in dealing with variation in size. The experimental results in Section 4.2 show that enforcing a hierarchical structure on the encoder boosts anomaly detection performance as well. For this reason, we employ a hierarchical encoder that embodies a smaller encoder; the hierarchical encoder is defined as

$$f_{\text{big}}(\mathbf{p}) = g_{\text{big}}(f_{\text{small}}(\mathbf{p})). \tag{7}$$

An input patch $\mathbf{p}$ is divided into a 2×2 grid, and their features are aggregated to constitute the feature of $\mathbf{p}$, as shown in Fig. 5. Each encoder with receptive field size K is trained with the self-supervised task of patch size K. Throughout the experiment, the receptive field sizes of the large and small encoders are set to 64 and 32, respectively.

3.4   Generating anomaly maps

After training the encoders, the representations from the encoder are used to detect the anomalies. First, the representation of every normal train patch, $\{f_\theta(\mathbf{p}_{\text{normal}})|\mathbf{p}_{\text{normal}}\}$, is calculated and stored. Given a query image $\mathbf{x}$, for every patch $\mathbf{p}$ with a stride S within $\mathbf{x}$, the L2 distance to the nearest normal patch in the feature space is then defined to be its anomaly score using Eq. 8. To mitigate the computational cost of the nearest neighbor search, we adopt its approximate algorithm[2]. As a result, the inspection of a single image from MVTec AD [8] dataset for example, requires approximately 0.48 second.

$$\mathcal{A}_\theta^{\text{patch}}(\mathbf{p}) \doteq \min_{\mathbf{p}_{\text{normal}}} \|f_\theta(\mathbf{p}) - f_\theta(\mathbf{p}_{\text{normal}})\|_2 . \tag{8}$$

---

[2] https://github.com/yahoojapan/NGT

Table 1: **Anomaly detection (left) and segmentation (right) performances on MVTec AD [8] dataset.** The proposed method, Patch SVDD, achieves the state-of-the-art performances on both tasks.

Method	AUROC		Method	AUROC
Task: Anomaly Detection			Task: Anomaly Segmentation	
Deep SVDD [4] ICML' 18	0.592		L2-AE	0.804
GEOM [19] NeurIPS' 18	0.672		SSIM-AE	0.818
GANomaly [15] ACCV' 18	0.762		VE VAE CVPR' 20 [18]	0.861
ITAE [17] arXiv' 19	0.839		VAE Proj ICLR' 20 [14]	0.893
Patch SVDD (Ours)	**0.921**		**Patch SVDD (Ours)**	**0.957**

Patch-wise calculated anomaly scores are then distributed to the pixels. As a consequence, pixels receive the average anomaly scores of every patch to which they belong, and we denote the resulting anomaly map as $\mathcal{M}$.

The multiple encoders discussed in Section 3.3 constitute multiple feature spaces, thereby yielding multiple anomaly maps. We aggregate the multiple maps using element-wise multiplication, and the resulting anomaly map, $\mathcal{M}_{\mathrm{multi}}$, provides the answer to the problem of anomaly segmentation:

$$\mathcal{M}_{\mathrm{multi}} \doteq \mathcal{M}_{\mathrm{small}} \odot \mathcal{M}_{\mathrm{big}}, \tag{9}$$

where $\mathcal{M}_{\mathrm{small}}$ and $\mathcal{M}_{\mathrm{big}}$ are the generated anomaly maps using f_{small} and f_{big}, respectively. The pixels with high anomaly scores in the map $\mathcal{M}_{\mathrm{multi}}$ are deemed to contain defects.

It is straightforward to address the problem of anomaly detection. The maximum anomaly score of the pixels in an image is its anomaly score, as expressed in Eq. 10. Fig. 6 illustrates the overall flow of the proposed method, and its pseudo-code is provided in the supplementary materials.

$$\mathcal{A}_{\theta}^{\mathrm{image}}(\mathbf{x}) \doteq \max_{i,j} \mathcal{M}_{\mathrm{multi}}(\mathbf{x})_{ij}. \tag{10}$$

4   Results and Discussion

To verify the validity of the proposed method, we applied it to MVTec AD [8] dataset. The dataset consists of 15-class industrial images, each class categorized as either an *object* or *texture*. Ten *object* classes contain regularly positioned objects, whereas the *texture* classes contain repetitive patterns. The implementation details used throughout the study are provided in the supplementary materials, and please refer to [8] for more details on the dataset.

4.1   Anomaly detection and segmentation results

Fig. 7 shows anomaly maps generated using the proposed method, indicating that the defects are properly localized, regardless of their size. Table 1 shows the
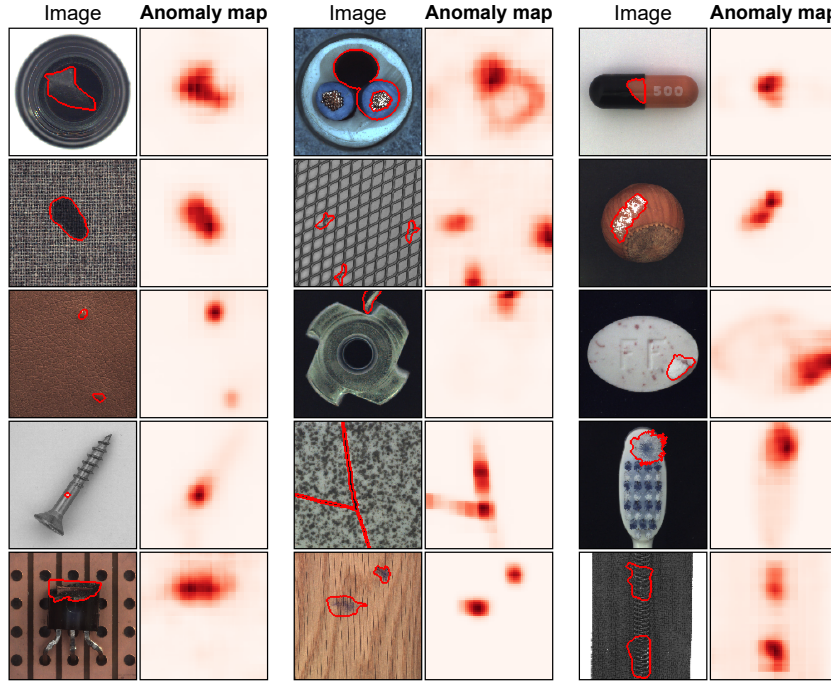
Fig. 7: **Anomaly maps generated by the proposed method.** Patch SVDD generates anomaly maps of each image in fifteen classes of MVTec AD [8] dataset. The ground truth defect annotations are depicted as red contours in the image, and the darker heatmap indicates higher anomaly scores.

detection and segmentation performances for MVTec AD [8] dataset compared with state-of-the-art baselines in AUROC. Patch SVDD provides state-of-the-art performance over the powerful baselines including auto encoder-based and classifier-based methods and outperforms Deep SVDD [4] by 55.6% improvement. Detailed numerical results are provided in the supplementary materials.

4.2   Detailed analysis

t-SNE visualization Fig. 8 shows t-SNE visualizations [30] of the learned features of multiple train images. Patches located at the points shown in Fig. 8(b) are mapped to the points with the same color and size in Fig. 8(a) and Fig. 8(c). In Fig. 8(a), the points with similar color and size form clusters in the feature space. Since the images in the cable class are regularly positioned, the patches from the same position have similar content, even if they are from different images. Likewise, for the regularly positioned *object* classes, the points with similar color and size in t-SNE visualization (i.e., the patches with similar positions) can be regarded to be semantically similar. By contrast, features of the leather class in Fig. 8(c) show the opposite tendency. This is because the patches in *texture*
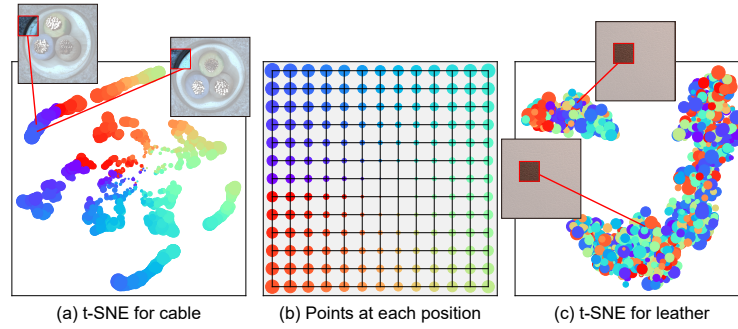
| (a) t-SNE for cable | (b) Points at each position | (c) t-SNE for leather |

Fig. 8: **t-SNE visualizations [30] of the learned features.** The color and size of each point represent the position (θ and r of the polar coordinates) within an image (b). From its color and size, we can infer the positions of the corresponding patches of the features in (a, c).

Table 2: **The effect of the losses.** Modifying $\mathcal{L}_{\text{SVDD}}$ to $\mathcal{L}_{\text{SVDD}'}$ and adopting $\mathcal{L}_{\text{SSL}}$ both improve the anomaly detection (`Det.`) and segmentation (`Seg.`) performances.

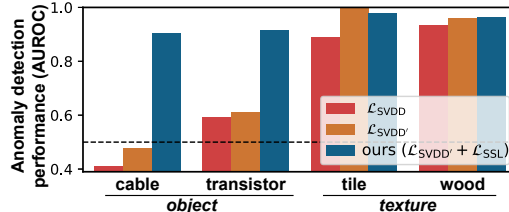| $\mathcal{L}_{\text{SVDD}}$	$\mathcal{L}_{\text{SVDD}'}$	$\mathcal{L}_{\text{SSL}}$	Det.	Seg.
✓	✗	✗	0.703	0.832
✗	✓	✗	0.739	0.880
✗	✓	✓	**0.921**	**0.957**

Fig. 9: **The effects of the losses vary among classes.** $\mathcal{L}_{\text{SSL}}$ is particularly beneficiary to the *object* classes.

classes are analogous, regardless of their position in the image; the positions of the patches are not quite related to their semantics for the *texture* images.

Effect of self-supervised learning Patch SVDD trains an encoder using two losses: $\mathcal{L}_{\text{SVDD}'}$ and $\mathcal{L}_{\text{SSL}}$, where $\mathcal{L}_{\text{SVDD}'}$ is a variant of $\mathcal{L}_{\text{SVDD}}$. To compare the roles of the proposed loss terms, we conduct an ablation study. Table 2 suggests that the modification of $\mathcal{L}_{\text{SVDD}}$ to $\mathcal{L}_{\text{SVDD}'}$ and the adoption of $\mathcal{L}_{\text{SSL}}$ improve the anomaly detection and segmentation performances. Fig. 9 shows that the effects of the proposed loss terms vary among classes. Specifically, the *texture* classes (e.g. tile and wood) are less sensitive to the choice of loss, whereas the *object* classes, including cable and transistor, benefit significantly from $\mathcal{L}_{\text{SSL}}$.

To investigate the reason behind these observations, we provide (in Fig. 10) t-SNE visualizations of the features of an *object* class (the transistor) for the encoders trained with $\mathcal{L}_{\text{SVDD}}$, $\mathcal{L}_{\text{SVDD}'}$, and $\mathcal{L}_{\text{SVDD}'} + \mathcal{L}_{\text{SSL}}$. When training is performed with $\mathcal{L}_{\text{SVDD}}$ (Fig. 10(a)) or $\mathcal{L}_{\text{SVDD}'}$ (Fig. 10(b)), the features form a uni-modal cluster. In contrast, $\mathcal{L}_{\text{SSL}}$ results in multi-modal feature clusters on the basis of their semantics (i.e., color and size), as shown in Fig. 10(c). The
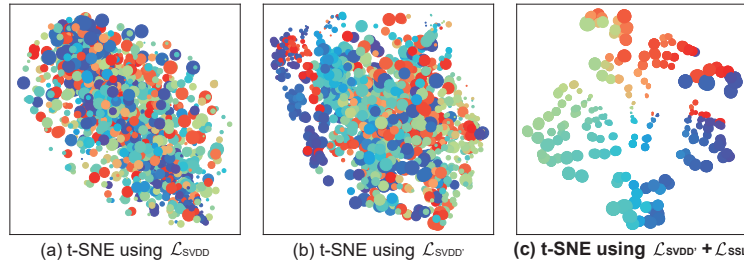
(a) t-SNE using $\mathcal{L}_{\text{SVDD}}$ (b) t-SNE using $\mathcal{L}_{\text{SVDD'}}$ (c) **t-SNE using $\mathcal{L}_{\text{SVDD'}} + \mathcal{L}_{\text{SSL}}$**

Fig. 10: **Features of the multiple train images in the transistor class by the encoders trained with different losses**. Adopting $\mathcal{L}_{\text{SSL}}$ (c) enables the representations to form clusters on the basis of their semantics.
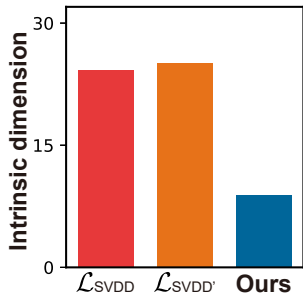

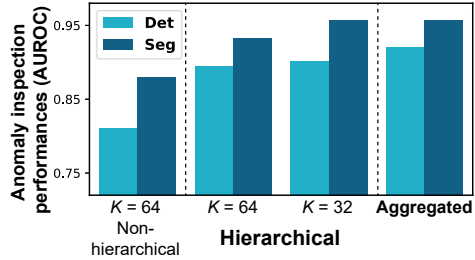Fig. 11: **Intrinsic dimensions of the features under different losses**.

Fig. 12: **The effect of hierarchical encoding.** Aggregating the results from multi-scale inspection boosts the performance, and adopting hierarchical structure to the encoder is helpful as well.

multi-modal property of the features is particularly beneficial to the *object* classes, which have high intra-class variation among the patches. Features of the patches with dissimilar semantics are separated, and hence anomaly inspection using those features becomes more deliberate and accurate.

The intrinsic dimensions (ID) [31] of the features also indicate the effectiveness of $\mathcal{L}_{\text{SSL}}$. The ID is the minimal number of coordinates required to describe the points without significant information loss [32]. A larger ID denotes that the points are spreaded in every direction, while a smaller ID indicates that the points lie on low-dimensional manifolds with high separability. In Fig. 11, we show the average IDs of features in each class trained with three different losses. If the encoder is trained with the proposed $\mathcal{L}_{\text{Patch SVDD}}$, features with the lowest ID are yielded, implying that these features are neatly distributed.

Hierarchical encoding In Section 3.3, we proposed the use of hierarchical encoders. Fig. 12 shows that aggregating multi-scale results from multiple encoders improves the inspection performances. In addition, an ablation study with a non-hierarchical encoder shows that the hierarchical structure itself also boosts performance. We postulate that the hierarchical architecture provides
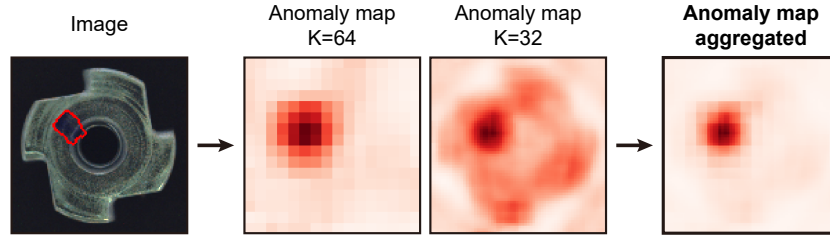
Fig. 13: **Multi-scale inspection.** Patch SVDD performs multi-scale inspection and aggregates the results. The image is from MVTec AD [8] dataset.

regularization for the feature extraction. Note that the non-hierarchical encoder has a number of parameters similar to that of the hierarchical counterpart.

We provide an example of multi-scale inspection results, together with an aggregated anomaly map, in Fig. 13. The anomaly maps from various scales provide complementary inspection results; the encoder with a large receptive field coarsely locates the defect, whereas the one with a smaller receptive field refines the result. Therefore, an element-wise multiplication of the two maps localizes the accurate position of the defect.

Hyperparameters As shown in Eq. 6, the hyperparameter λ balances $\mathcal{L}_{\text{SVDD}'}$ and $\mathcal{L}_{\text{SSL}}$. A large λ emphasizes gathering of the features, while a small λ promotes their informativeness. Interestingly, the most favorable value of λ varies among the classes. Anomalies in the *object* classes are well detected under a smaller λ, while the *texture* classes are well detected with a larger λ. Fig. 14 shows an example of this difference; the anomaly detection performance for the cable class (*object*) improves as λ decreases, while the wood class (*texture*) shows the opposite trend. As discussed in the previous sections, this occurs because the self-supervised learning is more helpful when the patches show high intra-class variation, which is the case for the *object* classes. The result coincides with that shown in Fig. 9 because using $\mathcal{L}_{\text{SVDD}'}$ as a loss is equivalent to using $\mathcal{L}_{\text{Patch SVDD}}$ with $\lambda >> 1$.

The number of feature dimensions, D, is another hyperparameter of the encoder. The anomaly inspection performance for varying D is depicted in Fig. 15(a). A larger D signifies improved performance—a trend that has been discussed in a self-supervised learning venue [28]. Fig. 15(b) indicates that the ID of the resulting features increases with increasing D. The black dashed line represents the $y = x$ graph, and it is the upper bound of ID. The average ID of features among the classes saturates as $D = 64$; therefore, we used a value of $D = 64$ throughout our study.

Random encoder Doersch et al. [6] showed that randomly initialized encoders perform reasonably well in image retrieval; given an image, the nearest images in the random feature space look similar to humans as well. Inspired by this
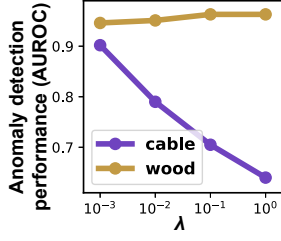
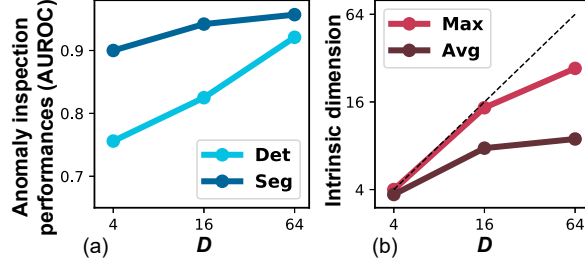Fig. 14: Anomaly detection performances for the two classes show the opposite trends as λ varies.

Fig. 15: **The effect of the embedding dimension, D.** Larger D yields better inspection results (a) and larger intrinsic dimensions (b).

observation, we examined the anomaly detection performance of the random encoders and provided the results in Table 3. As in Eq. 8, the anomaly score is defined to be the distance to the nearest normal patch, but in the random feature space. In the case of certain classes, the features of the random encoder are effective in distinguishing between normal and abnormal images. Some results even outperform the trained deep neural network model (L2-AE).

Here, we investigate the reason for the high separability of the random features. For simplicity, let us assume the encoder to be a one-layered convolutional layer parametrized by a weight $W \neq \mathbf{0}$ and a bias b followed by a nonlinearity, σ. Given two patches $\mathbf{p}_1$ and $\mathbf{p}_2$, their features h_1 and h_2 are provided by Eq. 11, where $*$ denotes a convolution operation.

$$
\begin{aligned}
h_1 &= \sigma(W * \mathbf{p}_1 + b) \\
h_2 &= \sigma(W * \mathbf{p}_2 + b).
\end{aligned}
\tag{11}
$$

As Eq. 12 suggests, when the features are close, so are the patches, and vice versa. Therefore, nearest patch in the feature space is likely to be so in the image space.

$$
\begin{aligned}
\|h_1 - h_2\|_2 \approx 0 &\Leftrightarrow (W * \mathbf{p}_1 + b) - (W * \mathbf{p}_2 + b) \approx 0 \\
&\Leftrightarrow W * (\mathbf{p}_1 - \mathbf{p}_2) \approx 0 \\
&\Leftrightarrow \|\mathbf{p}_1 - \mathbf{p}_2\|_2 \approx 0.
\end{aligned}
\tag{12}
$$

In Table 3, we also provide the results for anomaly detection task using the nearest neighbor algorithm using the raw patches (i.e., $f_\theta(\mathbf{p}) = \mathbf{p}$ in Eq. 8). For certain classes, the raw patch nearest neighbor algorithm works surprisingly well. The effectiveness of the raw patches for anomaly detection can be attributed to the high similarity among the normal images.

Furthermore, the well-separated classes provided by the random encoder are well-separated by the raw patch nearest neighbor algorithm, and vice versa. Together with the conclusion of Eq. 12, this observation implies the strong relationship between the raw image patch and its random feature. To summarize,

Table 3: **Nearest neighbor algorithm using the random encoders and raw patches for MVTec AD [8] dataset.** For certain classes, the nearest neighbor algorithm using the random features shows good anomaly detection performance. For those classes, using the raw patches also yields high performance.

Classes	Random Encoder		Raw Patch		L2-AE
	Det.	Seg.	Det.	Seg.	Seg.
bottle	0.89	0.80	**0.92**	0.82	0.91
cable	0.56	0.70	0.57	0.84	0.73
capsule	0.67	**0.93**	0.67	**0.94**	0.79
carpet	0.42	0.66	0.48	0.74	0.54
grid	0.75	0.61	0.83	0.76	0.96
hazelnut	0.83	**0.93**	0.83	**0.95**	0.98
leather	0.69	0.85	0.69	0.88	0.75
metal_nut	0.37	0.79	0.52	0.89	0.88
pill	0.72	**0.92**	0.77	**0.92**	0.89
screw	0.69	**0.93**	0.56	**0.93**	0.98
tile	0.68	0.54	0.70	0.51	0.48
toothbrush	0.81	**0.95**	**0.92**	**0.97**	0.97
transistor	0.53	0.88	0.68	**0.92**	0.91
wood	**0.90**	0.74	**0.94**	0.78	0.63
zipper	0.70	0.84	**0.94**	**0.92**	0.68

the random features of anomalies are easily separable because they are alike the raw patches, and the raw patches are easily separable.

5   Conclusion

In this paper, we proposed Patch SVDD, a method for image anomaly detection and segmentation. Unlike Deep SVDD [4], we inspect the image at the patch level, and hence we can also localize defects. Moreover, additional self-supervised learning improves detection performance. As a result, the proposed method achieved state-of-the-art performance on MVTec AD [8] dataset.

In previous studies [23,24], images were featurized prior to the subsequent downstream tasks because of their high-dimensional and structured nature. However, the results in our analysis suggest that a nearest neighbor algorithm with a raw patch often discriminates anomalies surprisingly well. Moreover, since the distances in random feature space are closely related to those in the raw image space, random features can provide distinguishable signals.

6   Acknowledgement

References

1. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural computation **13** (2001) 1443–1471
2. Tax, D.M., Duin, R.P.: Support vector data description. Machine learning **54** (2004) 45–66
3. Vert, R., Vert, J.P.: Consistency and convergence rates of one-class svms and related algorithms. Journal of Machine Learning Research **7** (2006) 817–854
4. Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S.A., Binder, A., Müller, E., Kloft, M.: Deep one-class classification. In: ICML. (2018)
5. Ruff, L., Vandermeulen, R.A., Görnitz, N., Binder, A., Müller, E., Müller, K.R., Kloft, M.: Deep semi-supervised anomaly detection. In: ICLR. (2020)
6. Doersch, C., Gupta, A., Efros, A.A.: Unsupervised visual representation learning by context prediction. In: ICCV. (2015)
7. Minderer, M., Bachem, O., Houlsby, N., Tschannen, M.: Automatic shortcut removal for self-supervised representation learning. In: ICML. (2020)
8. Bergmann, P., Fauser, M., Sattlegger, D., Steger, C.: Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection. In: CVPR. (2019)
9. Calders, T., Jaroszewicz, S.: Efficient auc optimization for classification. In: European Conference on Principles of Data Mining and Knowledge Discovery, Springer (2007) 42–53
10. Lu, Y., Xu, P.: Anomaly detection for skin disease images using variational autoencoder. arXiv preprint arXiv:1807.01349 (2018)
11. Sabokrou, M., Khalooei, M., Fathy, M., Adeli, E.: Adversarially learned one-class classifier for novelty detection. In: CVPR. (2018)
12. Perera, P., Nallapati, R., Xiang, B.: Ocgan: One-class novelty detection using gans with constrained latent representations. In: CVPR. (2019)
13. Bergmann, P., Löwe, S., Fauser, M., Sattlegger, D., Steger, C.: Improving unsupervised defect segmentation by applying structural similarity to autoencoders. arXiv preprint arXiv:1807.02011 (2018)
14. Dehaene, D., Frigo, O., Combrexelle, S., Eline, P.: Iterative energy-based projection on a normal data manifold for anomaly localization. In: ICLR. (2020)
15. Akcay, S., Atapour-Abarghouei, A., Breckon, T.P.: Ganomaly: Semi-supervised anomaly detection via adversarial training. In: ACCV. (2018)
16. Yarlagadda, S.K., Güera, D., Bestagini, P., Maggie Zhu, F., Tubaro, S., Delp, E.J.: Satellite image forgery detection and localization using gan and one-class classifier. Electronic Imaging **2018** (2018) 214–1
17. Huang, C., Cao, J., Ye, F., Li, M., Zhang, Y., Lu, C.: Inverse-transform autoencoder for anomaly detection. arXiv preprint arXiv:1911.10676 (2019)
18. Liu, W., Li, R., Zheng, M., Karanam, S., Wu, Z., Bhanu, B., Radke, R.J., Camps, O.: Towards visually explaining variational autoencoders. In: CVPR. (2020)
19. Golan, I., El-Yaniv, R.: Deep anomaly detection using geometric transformations. In: NeurIPS. (2018)
20. Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. In: ICLR. (2017)
21. Wang, S., Zeng, Y., Liu, X., Zhu, E., Yin, J., Xu, C., Kloft, M.: Effective end-to-end unsupervised outlier detection via inlier priority of discriminative network. In: NeurIPS. (2019)

22. Bergman, L., Hoshen, Y.: Classification-based anomaly detection for general data. In: ICLR. (2020)
23. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: ECCV. (2016)
24. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: European conference on computer vision, Springer (2016) 649–666
25. Noroozi, M., Pirsiavash, H., Favaro, P.: Representation learning by learning to count. In: ICCV. (2017)
26. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. In: ICLR. (2018)
27. Koch, G., Zemel, R., Salakhutdinov, R.: Siamese neural networks for one-shot image recognition. In: ICML deep learning workshop. Volume 2., Lille (2015)
28. Kolesnikov, A., Zhai, X., Beyer, L.: Revisiting self-supervised visual representation learning. In: CVPR. (2019) 1920–1929
29. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR. (2015)
30. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9** (2008) 2579–2605
31. Facco, E., d'Errico, M., Rodriguez, A., Laio, A.: Estimating the intrinsic dimension of datasets by a minimal neighborhood information. Scientific reports **7** (2017) 1–8
32. Ansuini, A., Laio, A., Macke, J.H., Zoccolan, D.: Intrinsic dimension of data representations in deep neural networks. In: NeurIPS. (2019)