SuperADD: Training-free Class-agnostic Anomaly Segmentation – CVPR 2026 VAND 4.0 Workshop Challenge Industrial Track

Lukas Roming[1] Felix Lehnerer[1] Jonas V. Funk[2] Andreas Michel[1]
Georg Maier[1] Thomas Längle[1] Jürgen Beyerer[1]

[1]Fraunhofer IOSB, Fraunhoferstr. 1, 76131 Karlsruhe, Germany
[2]Independent Research
lukas.roming@iosb.fraunhofer.de

Abstract

Visual anomaly detection (AD) for industrial inspection is a highly relevant task in modern production environments. The problem becomes particularly challenging when training and deployment data differ due to changes in acquisition conditions during production. In the VAND 4.0 *Industrial Track, models must remain robust under distribution shifts such as varying illumination and their performance is assessed on the MVTec AD 2 dataset. To address this setting, we propose a training-free and class-agnostic anomaly detection pipeline based on the work of SuperAD. Our approach improves generalization through several modifications designed to enhance robustness under distribution shifts. These adaptations include using a DINOv3 backbone, overlapping patch-wise processing, intensity-based augmentations, improved memory-bank subsampling for better coverage of the data distribution, and iterative morphological closing for cleaner and more spatially consistent anomaly maps. Unlike methods that rely on class-specific architectures or per-class hyperparameter tuning, our method uses a single architecture and one shared hyperparameter configuration across all object classes. This makes the approach well suited for industrial deployment, where product variants and appearance changes must be handled with minimal adaptation effort. We achieve segmentation* F_1 *scores of* 62.61%, 57.42%, *and* 54.35% *on* TEST_{pub}, TEST_{priv}, *and* $\text{TEST}_{priv,mix}$ *of MVTec AD 2, respectively, thereby surpassing SuperAD and other state-of-the-art methods. Code is available at* https://github.com/LukasRoom/SuperADD.

1. Introduction

1.1. Background

Automated visual anomaly detection (AD) is increasingly used in industrial inspection, where high-throughput production and subtle defects create demanding operating conditions. Despite this growing adoption, performance often degrades under varying acquisition conditions, particularly illumination shifts that alter object appearance without affecting semantic structure [1]. This sensitivity motivates the need for the *VAND 4.0* Industrial Track, which focuses on robustness to such variations while maintaining real-time efficiency.

1.2. Challenge Description

The *VAND 4.0* Industrial Track evaluates AD models under acquisition shifts. Models are trained exclusively on normal images and must detect anomalies at test time. The task emphasizes balancing detection performance with computational efficiency.

Dataset: The challenge uses the MVTec Anomaly Detection 2 (MVTec AD 2) dataset [2], a benchmark for unsupervised AD [3]. The dataset comprises eight real-world classes captured under varying lighting conditions, leading to appearance shifts between training and test data. The absence of openly available ground truth for the private test split further increases its resemblance to real-world conditions by enforcing blind evaluation and reducing the risk of overfitting. The eight object classes can be grouped according to visual characteristics (non-exclusive):

- Bulk goods objects: *wall plugs*, *walnuts*, and *rice*, featuring overlapping and occluded instances arranged in uncontrolled spatial patterns.
- Textured objects: *fabric* and *sheet metal*, exhibiting high variability in normal appearance.

- Reflective metal objects: *sheet metal* and *can*.
- Transparent objects: *vial* and *fruit jelly*, which are particularly sensitive to illumination changes.

In addition, a range of lighting conditions is applied to the private split ($TEST_{\text{priv}}$), resulting in an additional split further amplifying distribution shifts between training and test data ($TEST_{\text{priv,mix}}$).

Challenge Goal: Methods are evaluated by their pixel-level F_1 score and ranked separately on $TEST_{\text{priv}}$ and $TEST_{\text{priv,mix}}$ of MVTec AD 2. The final score is the mean rank across these two test sets. In contrast to *VAND 3.0*, where successful methods often relied on class-specific architectures or class-specific hyperparameter tuning [3], this year's focus is on a single class-agnostic solution that generalizes across diverse objects. To enforce this, proposed solutions must use the same architecture and hyperparameters across classes.

1.3. Related Work

The original MVTec AD benchmark [4] has become largely saturated, with state-of-the-art methods achieving very high segmentation performance and leaving little room for meaningful comparison. This limits its usefulness for distinguishing between recent AD approaches and motivates the development of more challenging benchmarks such as MVTec AD 2.

Among the wide range of unsupervised AD methods, the strongest solutions in last year's *VAND 3.0* challenge (ISVL [5], RoBiS [6]) were largely based on INP-Former [7]. INP-Former is a reconstruction-based AD method that requires training to extract intrinsic normal prototypes from the input image via a Vision Transformer and reconstructs only the normal content. Regions that cannot be reconstructed well are treated as anomalies, and the reconstruction residual provides the anomaly score.

In contrast, we follow the memory bank approach of SuperAD [8], inspired by PatchCore [9], adopting a training-free strategy that avoids model retraining. In SuperAD, normal images are used to build a memory bank using the feature extraction capabilities of DINOv2 [10].

We build on this architecture because its training-free design is easily adaptable and provides an attractive alternative to methods that require dedicated retraining. This makes the method particularly appealing for industrial deployment, where new product classes or design changes often need to be integrated quickly.

1.4. Our Contributions

While SuperAD already performed strongly in last year's challenge, we focus on improving the underlying architecture through generalizable modifications. Our goal is a robust and practically deployable AD pipeline that remains effective across diverse object classes. Our main contributions are as follows:

- We maintain a class-agnostic setup, using the same architecture and hyperparameters across all object classes.
- We improve the subsampling stage in memory bank construction for computational efficiency and better data distribution coverage.
- We introduce overlapping patch-wise preprocessing to reduce grid-position sensitivity and improve generalization.
- We apply a simple iterative morphological closing step in postprocessing to obtain more spatially consistent anomaly maps.
- We employ intensity-based augmentations to simulate varying illumination conditions, improving robustness to lighting changes between training and test data.
- We replace the original DINOv2 backbone with DINOv3, leveraging improved pre-trained visual representations.

2. Methodology

2.1. Model Design

2.1.1. Approach

We build upon SuperAD from Zhang et al. [8]. We chose this architecture for its training-free and flexible design enabling faster development, hyperparameter tuning, and deployment. Just like SuperAD [8] and PatchCore [9], we create a memory bank at multiple intermediate layer outputs of a powerful visual feature extractor and perform nearest-neighbor-based outlier-detection at inference. As feature extractor, we employ DINOv3 [11] due to its state-of-the-art performance on a range of downstream classification and segmentation benchmarks.

To improve computational efficiency, we adopt a refined subsampling strategy. SuperAD applies PatchCore's coreset-reduction method to class tokens to select 16 images from which to extract memory-bank feature vectors. While this promotes diversity at the image level, it does not effectively eliminate near-duplicate feature vectors within the same image or similar regions appearing in different images, such as those arising from homogeneous background regions or from multiple, low-variation instances of the same object.

To address this limitation, we perform subsampling directly on the feature vectors rather than on the images. Specifically, we use a nearest-neighbor-based subsampling procedure with the same objective as the coreset-reduction approach of PatchCore, to remove near-duplicate embeddings from the memory bank. Details on the subsampling are given in section 2.1.3.

Due to the position invariance of several classes (*rice*, *walnuts*, and *wall plugs*), we adopted patch-wise image processing to reduce position-dependent artifacts. These artifacts include false AD in empty regions that the model

was not trained on. Patch-wise inference limits the model's global view and thus its sensitivity to such cases, as seen for wall plugs. Patch-wise processing improved performance for most classes (except fabric and vial).

To enhance the robustness of AD, we further apply intensity scaling to the training images to simulate variations in integration time.

In the original SuperAD implementation, class-specific thresholds are derived from test data with its ground truth by selecting the threshold that maximizes F_1 score. Instead, we define the threshold as a scaled version of the 95th percentile of the anomaly map values observed in the training data. This improves practical usability and complies with the *VAND 4.0* guidelines.

2.1.2. Architecture

An overview of the proposed method is illustrated in Figure 1. The main components are patching, a vision transformer backbone, and k-nearest neighbors comparison.

Overlapping patches: The input image of shape $H \times W$ is divided into equally sized patches of size $P \times P$. The number of patches n_d along a dimension d of size L_d with a minimum overlap of size O is then determined by (1). The start positions of patches are then spread linearly between 0 and $L_d - P$ such that the first patch starts at 0 and the last patch ends at $L_d - 1$. This guarantees a minimum overlap between patch borders inside the image. The overlap between patches reduces the vulnerability to artifacts occurring at the edges of the inference window. The alignment with image borders eliminates the need for padding which would lead to unrealistic reference embeddings. For example, zero padding could result in incomplete objects being labeled as non-anomalous. We chose a patch size of $P = 640$ with a minimum overlap of $O = 128$.

$$n_d = \left\lceil \frac{L_d - P}{P - 2O} \right\rceil + 1, \tag{1}$$

Backbone: We adopt the pretrained model DINOv3-ViT-H+/16 consisting of 32 transformer layers (or blocks) with approx. 840 million parameters. We extract embeddings from the layers 7, 15, 23, and 31. After inference, redundant predictions in the overlapping regions at patch borders are discarded. The remaining embedding vectors are then mapped back to their original spatial positions and concatenated, producing a single coherent inference result for the whole image.

Deriving anomaly maps: Following SuperAD [8], we perform nearest-neighbor search for the selected layer outputs and average across the nearest-neighbor distance maps. Finally, we post-process the floating point anomaly maps using thresholding, closing, and a fill region method as described in section 2.1.5.

2.1.3. Training

Just like SuperAD, our method does not have any parameters to be trained and all model weights stay fixed. We use the training data to extract prototype embeddings from four layers (as described in section 2.1.2) and derive an anomaly threshold.

Pre-processing: First, images are downscaled by a factor of 0.625 and normalized using ImageNet normalization (this is also used for inference). Normalized pixel values are then scaled by a random factor uniformly sampled from the interval $[0.8, 1.2]$. This is done to improve robustness to the distribution shift caused by varying lighting conditions.

Prototype embedding subsampling: Instead of selecting 16 images of the training samples for prototype generation as suggested by [8], we use all available training images (reserving $1/8$ of them for threshold estimation) and apply a fast subsampling scheme to construct a compact and diverse memory bank. This reduces memory usage and inference time while removing redundant prototypes.

Specifically, we perform a k-nearest-neighbor-based selection in feature space:

1. For each candidate feature vector, compute the distances to its $k = 100$ nearest neighbors.
2. Compute a global distance threshold τ as the mean of all k-NN distances.
3. For each feature vector, define a subsampling score as the number of its k neighbors whose distance is smaller than τ. Vectors with low scores lie in sparsely populated regions of the feature space and are therefore more informative.
4. Sort all feature vectors by this subsampling score in ascending order and retain the first n vectors, where n is the target memory-bank size.

To further accelerate the procedure, it can be applied independently to several (random) subsets of the original feature set. The selected prototypes from all subsets are then merged. Empirically, this subset-wise subsampling not only reduces computation time but can also yield equal or slightly improved mean F_1 scores compared to running the algorithm once on the full feature set.

Threshold-estimation: We process $1/8$ of the training samples (that were not used for prototype generation) the same way as we do the inference of test samples, except that we stop at the floating point valued distance map. We therefore, compare those separated training samples to the prototype embeddings in the memory bank and calculate the nearest-neighbor distance map. Using the values of the map, we define the threshold such that it splits the 95th percentile. We further scale this threshold by a constant gain factor, whereby a gain in the range of 1.3 to 1.5 worked well in our experiments. We found that his strategy is more robust and yields better results than using a higher percentile (e.g. 99th percentile) without additional scaling.

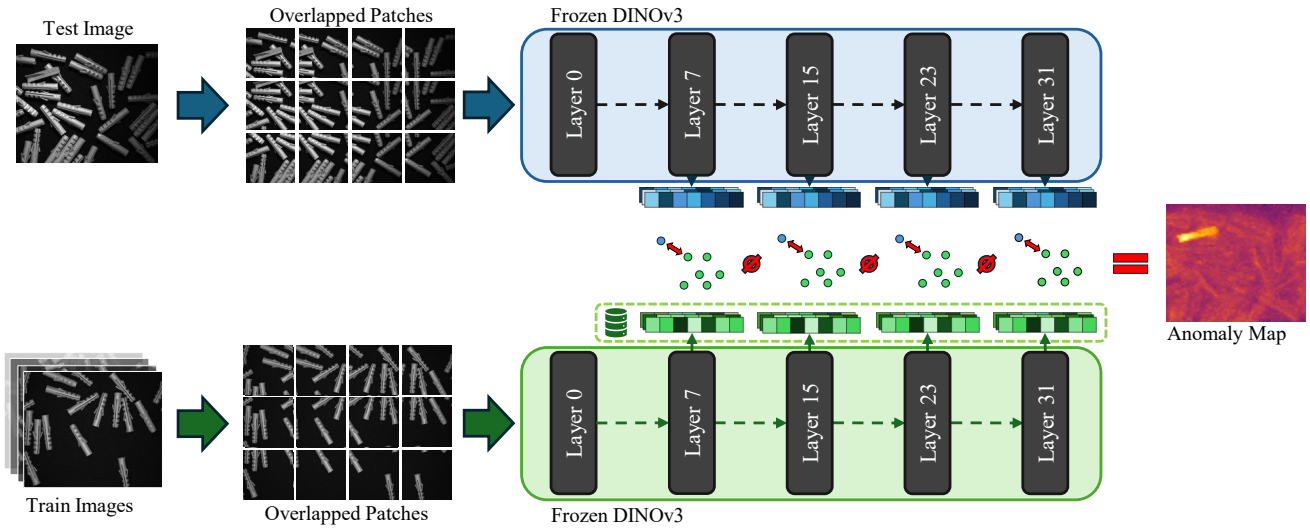

Figure 1. Overview of the proposed method to calculate anomaly maps based on SuperAD [8]. Inference is performed on patches for test and train images. Embeddings are extracted from multiple layers to create prototypes representing anomaly-free data. The anomaly score is determined by the average distance between a test embedding vector to its nearest neighbor prototype.

2.1.4. Inference

For inference of test images, we use the same model to compute embeddings and measure the distance to the nearest neighbor in the prototype memory bank. Distances are computed per 16×16 patch, corresponding to the model's internal token size (not to be confused with the 640×640 patches used at a higher level of the architecture). Following PatchCore, this procedure is applied to multiple intermediate DINOv3 layers: for each of the four selected layers, we construct a memory bank, compute a distance map for the test image, and then average the resulting maps.

The final anomaly map has a spatial resolution of $^{10}/_{256}$ of the original input image. This map is then upscaled to $^{1}/_{4}$ of the original resolution and subsequently post-processed.

2.1.5. Post-processing

Figure 2 shows an overview of the applied post-processing procedure. After the distance based anomaly scores are obtained a thresholding operation is applied with an automatically determined threshold value.

Morphological closing is applied to close small gaps in the detected binary masks to compensate for small false-negative detections at object or anomaly borders. A multi-oriented linear closing approach is applied to connect any broken linear features such as edges of anomalies. Morphological closing is performed 16 times using line structuring elements with a radius of 26 pixels and evenly spaced orientations between 0° and 180°. The results of the individual closing operations are combined with a pixelwise logical OR operation. To suppress unwanted artifacts the input image is zero-padded with $26 + 1$ pixels, and the padding is removed afterwards.

The result of morphological closing is masked using a binary mask obtained by thresholding the anomaly map at $0.8\times$ the determined anomaly threshold, to prevent oversegmentation.

Due to the preceding patching step, the resulting binary mask may contain holes where anomalies extend across patch boundaries. In a final postprocessing step closed regions are filled to eliminate any holes.

2.2. Dataset & Evaluation

2.2.1. Dataset Utilization

The MVTec AD 2 test data is divided into three subsets: $TEST_{\text{pub}}$, $TEST_{\text{priv}}$, and $TEST_{\text{priv,mix}}$. Ground-truth annotations are only available for $TEST_{\text{pub}}$, on which we perform local benchmarking. For $TEST_{\text{priv}}$ and $TEST_{\text{priv,mix}}$, only the images are publicly provided and final scores were obtained via the official evaluation server. $TEST_{\text{priv}}$ contains images captured under the same lighting conditions as the training set, whereas $TEST_{\text{priv,mix}}$ shows the same scenes but with a mixture of seen and unseen lighting conditions. For details about preprocessing and augmentations refer to section 2.1.2. The usage of training data is explained in section 2.1.3.

2.2.2. Evaluation Criteria

Results of the described method are evaluated following the challenge guidelines. We therefore report the pixel-wise F_1 score (2) on the public test dataset as it best aligns with the rating criteria of the challenge. Additionally the AU-ROC$_{0.05}$ scores are provided. Obtained scores are

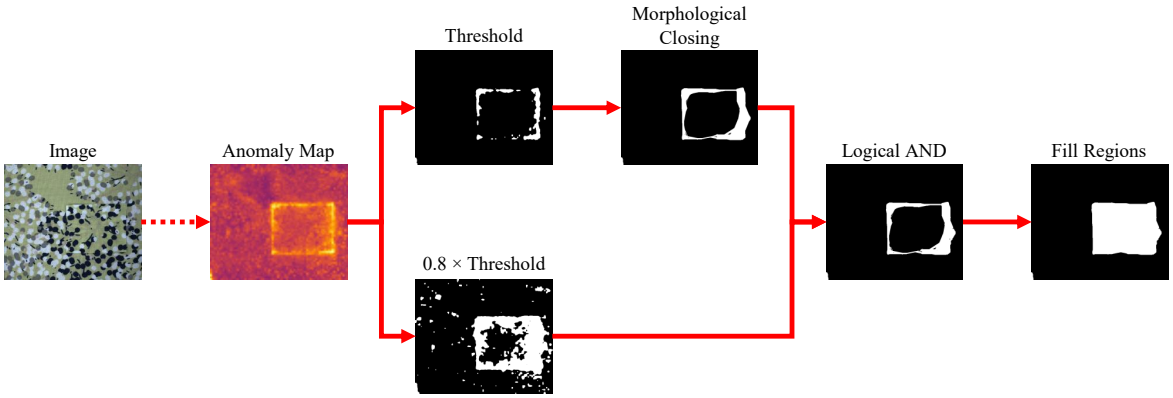

Figure 2. Overview of the postprocessing procedure. After initial thresholding, morphological closing is applied. The output is masked using a logical AND operation. Finally, closed regions are filled.

solely used to assess the quality of detection but not used to automatically determine optimal class-specific thresholds.

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{2}$$

2.2.3. Evaluation Setting

Our work focuses on the regular setting of the industrial track. For each category, we adapt a separate model instance using only the official anomaly-free training split. All model hyperparameters and thresholds are either static or selected automatically from training data in accordance with the challenge rules. The complete implementation, including all dependencies and a step-by-step guide to reproduce our results, is available at https://github.com/LukasRoom/SuperADD.

3. Results

3.1. Quantitative Results

We report the F_1 score and AU-ROC$_{0.05}$ for $TEST_{\text{pub}}$ in Table 2. We achieved high AU-ROC and F_1 scores for all classes except *can*. Samples of *can* in $TEST_{\text{pub}}$ contained fine defects that are barely visible to the human eye. The method achieved the highest F_1 score for *fabric* (93.74%), followed by *wallplugs* (79.16%).

Further, we report the F_1 scores for $TEST_{\text{priv}}$ and $TEST_{\text{priv,mix}}$ in Table 2. The highest F1-Scores of our approach for $TEST_{\text{priv}}$ were achieved for *fabric* (88.47%), matching the observation in $TEST_{\text{pub}}$, followed by *rice* (73.83%). Interestingly, performance on *wallplugs* is here significantly lower compared to $TEST_{\text{pub}}$, which can be attributed to the presence of substantially more difficult wallplug samples in $TEST_{\text{priv}}$, characterized by more subtle defects and likely a reduced tolerance for false positives due to the low number of positive instances in the ground truth.

We compare our results in Table 2 to the results of PatchCore, EfficientAD, and the top four approaches of last years challenge for $TEST_{\text{priv}}$ and $TEST_{\text{priv,mix}}$. We achieved top F_1 scores for several categories, including *rice* and *vial*. Most importantly, we outperformed all other methods in terms of mean F_1 score. We achieved 57.42% and 54.35% for $TEST_{\text{priv}}$ and $TEST_{\text{priv,mix}}$ respectively, compared to last years best (ISVL [5]) with 53.81% and 51.43% .

Object	**AU-ROC$_{0.05}$**	**F_1 score**
Can	51.61	0.00
Fabric	84.20	93.74
Fruit Jelly	81.19	54.68
Rice	96.11	73.31
Sheet Metal	93.18	59.54
Vial	82.59	64.77
Wallplugs	92.53	79.16
Walnuts	90.03	75.69
Mean	83.93	62.61

Table 1. Segmentation AU-ROC$_{0.05}$ and F_1 score (in %) for the proposed method on binarized images for $TEST_{\text{pub}}$.

3.2. Qualitative Results

We provide qualitative results for $TEST_{\text{pub}}$ in Figure 3. Some presented anomalies are very tiny and need close inspection to be identified including those of categories *can*, *rice*, and *sheet metal*.

For the *sheet metal* example shown in Figure 3, the model successfully detects the scratch. However, due to a suboptimal threshold, only a portion of it is retained in the thresholded map. The closing operation in the postprocessing ensures that the preserved part of the scratch forms a well-connected region.

Category	PatchCore [9]	EfficientAD [12]	ISVL [5]	RoBiS [6]	ASEG [13]	SuperAD [8]	SuperADD (ours)
Can	0.3 / 0.1	0.8 / 0.1	17.03 / 9.21	1.86 / 0.84	**18.14** / **10.13**	17.3 / 1.9	11.59 / 0.38
Fabric	11.5 / 9.8	7.6 / 1.0	84.95 / 81.76	87.46 / 73.37	46.53 / 31.74	77.4 / 65.3	**88.47** / **86.14**
Fruit Jelly	8.7 / 8.2	20.8 / 18.2	**68.1** / **67.76**	53.63 / 52.62	62.59 / 62.05	41.3 / 40.9	68.03 / 67.75
Rice	3.8 / 4.2	15.0 / 0.5	66.34 / 66.93	63.86 / 63.23	72.5 / 73.74	60.9 / 61.2	**73.83** / **74.32**
Sheet Metal	1.8 / 1.1	9.3 / 3.8	69.69 / 69.74	**70.98** / **70.92**	64.55 / 62.32	59.5 / 59.7	55.74 / 55.47
Vial	2.3 / 2.2	30.5 / 26.5	44.01 / 42.12	48.73 / 48.83	42.67 / 43.96	42.8 / 40.8	**64.37** / **63.52**
Wallplugs	0.0 / 0.0	4.4 / 0.3	11.99 / 6.24	14.38 / 3.40	5.5 / 5.06	13.7 / 6.7	**26.43** / **18.38**
Walnuts	1.2 / 1.3	34.6 / 13.3	68.34 / 67.65	67.13 / 58.94	67.49 / 67.21	69.1 / **69.1**	**70.88** / 68.84
Mean	3.7 / 3.4	15.4 / 8.0	53.81 / 51.43	51.00 / 46.52	47.5 / 44.49	47.8 / 43.2	**57.42** / **54.35**

Table 2. Performance comparison of segmentation F_1 score (in %) on binarized images for $TEST_{priv}$ / $TEST_{priv,mix}$ set. Bold values indicate the best scores.

An observed limitation of the proposed approach is the detection of missing parts in an image, such as broken pieces in *wallplugs* or an empty *vial*. Similarly, very thin scratches on *can* or *sheet metal* as well as a single hair in *fruit jelly* are often not reliably identified as anomalies. This drawback, however, does not apply to small defects in general: for categories such as *rice* or *walnut*, the model successfully localizes even very small anomalies. In contrast, large-scale defects are consistently detected with high accuracy, for example the fabric cutout placed on top of the base material in the *fabric* category, which is clearly and coherently segmented as an anomalous region, owing to the applied post-processing.

4. Discussion

4.1. Challenges & Solutions

During development, we identified several challenges and adapted our method accordingly.

Patch-border artifacts: Patch-wise inference with non-overlapping windows led to better overall performance, but at the same time to artifacts at patch borders, including discontinuities in the anomaly map and false detections in regions that were only partially visible within a patch. To mitigate these issues, we introduced overlapping patches in both spatial dimensions. Patch start positions are chosen such that patches are equally distributed and aligned with image borders, guaranteeing a minimum overlap without requiring padding. After inference, redundant predictions in overlapping regions are discarded when reassembling the full-image anomaly map.

Disconnected linear anomalies: For categories with thin, elongated defects (e.g. scratches on sheet metal), the raw anomaly maps and initial thresholding often produced fragmented segmentations with gaps along the defect. We address this by applying a multi-oriented morphological closing step to the thresholded anomaly map. This connects broken line segments along arbitrary directions while preserving the overall defect shape.

Large unfilled regions: Following the strategy proposed in SuperAD [8], we include a final hole-filling step. After morphological closing, the resulting mask is intersected with a slightly more permissive binary mask obtained by thresholding the anomaly map at 0.8 times the learned threshold, which constrains the fill operation and reduces over-selection. Finally, we fill closed regions in the binary mask, yielding more coherent anomaly segments and improving robustness to under-segmentation at patch boundaries.

4.2. Model Robustness & Adaptability

The model robustness is shown by the performance on the split $TEST_{\text{priv,mix}}$ compared to $TEST_{\text{priv}}$ in Table 2. It can be seen that there is a performance drop (overall $\approx$ 2 percentage points), most notably for the class can ($\approx$ 12 percentage points). But overall, the model still achieved the best performance on the more challenging $TEST_{\text{priv,mix}}$ split with 54.35%.

Moreover, the training-free design of our approach contributes to its robustness. Unlike fully trained deep learning models, it does not require potentially unstable optimization procedures, which are prone to convergence failures and often necessitate hyperparameter tuning to establish a reliable training scheme.

4.3. Future Work

One promising future direction is to extend this approach to a dual memory bank framework, similar to [14]. A possible strategy would be to synthesize anomalous images with a diffusion model by transferring defects from MVTec AD 1 onto normal images from MVTec AD 2. The feature alignment strategy of [15] could then be used to reduce the gap between synthetic and real defects, enabling the construction of a second, anomaly-focused memory bank.

The two memory banks could be used in a complementary manner: the normal branch would capture deviations from the normal distribution, while the anomalous branch would provide additional evidence of anomaly by matching

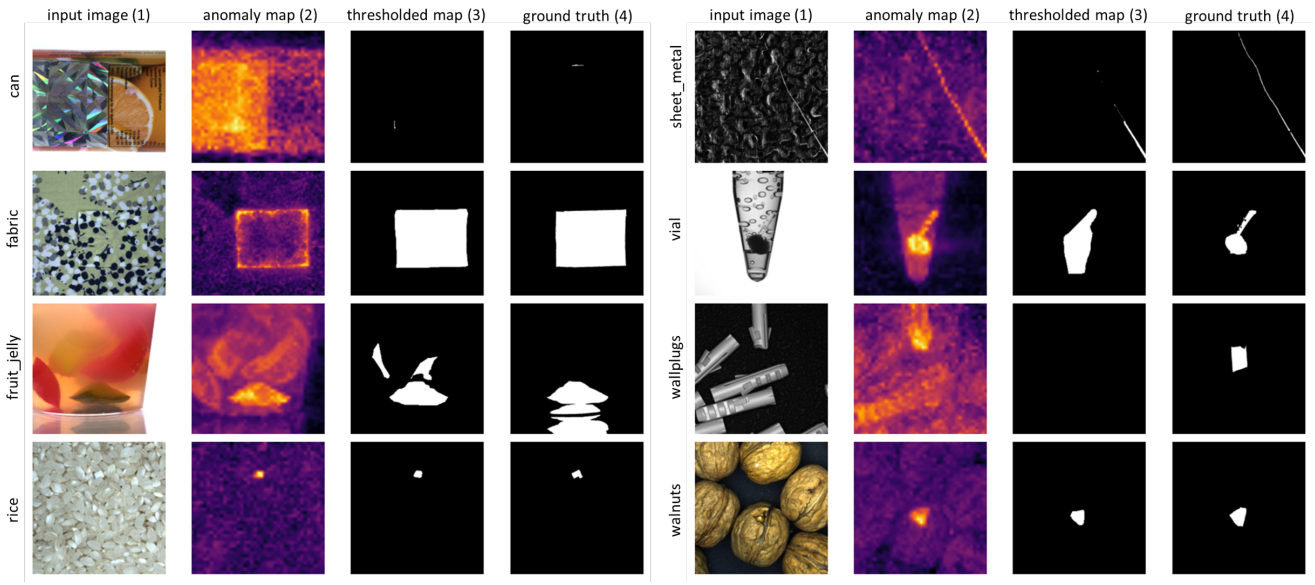

Figure 3. Qualitative results of the proposed method. Columns: (1) the input image, (2) the anomaly map, i.e., the real-valued anomaly scores obtained from the averaged distance maps before post-processing, (3) the binary anomaly map after post-processing, including thresholding, and (4) the ground-truth annotation.

features to synthetic defect prototypes. This direction remains training-free and could further enhance unsupervised AD performance.

5. Conclusion

In this paper, we propose a training-free, class-agnostic method for anomaly segmentation. Our method builds upon SuperAD [8], constructs multi-layer memory banks using DINOv3 as feature extractor, and performs nearest-neighbor-based outlier detection at inference. We introduce a refined subsampling strategy, employ patch-wise image processing, and derive thresholds solely from anomaly-free samples in the training data using the 95th percentile. We achieved pixel-level F_1 scores of 57.42% and 54.35% for $TEST_{\text{priv}}$ and $TEST_{\text{priv,mix}}$ respectively, compared to last years best (ISVL [5]) with 53.81% and 51.43% .

Our experiments demonstrate that combining the latest vision foundation model with patch-wise processing, improved subsampling, and post-processing significantly enhances the performance of training-free anomaly detection, achieving generalizable results competitive with state-of-the-art approaches.

References

[1] Zhuo Li, Yuhao Yan, Xiangheng Wang, Yifei Ge, and Lin Meng. A survey of deep learning for industrial visual anomaly detection. *Artificial Intelligence Review*, 58(9):279, 2025. 1

[2] Lars Heckler-Kram, Jan-Hendrik Neudeck, Ulla Scheler, Rebecca König, and Carsten Steger. The mvtec ad 2 dataset: Advanced scenarios for unsupervised anomaly detection. *International Journal of Computer Vision*, 134(4):175, 2026. 1

[3] Lars Heckler-Kram, Ashwin Vaidya, Jan-Hendrik Neudeck, Ulla Scheler, Dick Ameln, Samet Akcay, and Paula Ramos. From benchmarks to reality: Advancing visual anomaly detection by the vand 3.0 challenge. *arXiv preprint arXiv:2509.17615*, 2025. 1, 2

[4] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9592–9600, 2019. 2

[5] Xingao Wang, Shuying Xia, Zhaohong Liao, Mengjie Xie, Handa Wang, and Zhi Gao. Accurate anomaly localization in challenging industrial settings via a hybrid detection framework. In *Proceedings of the CVPR Workshop on Adapt & Detect (Track I)*. Wuhan University, 2025. 2, 5, 6, 7

[6] Xurui Li, Zhonesheng Jiang, Tingxuan Ai, and Yu Zhou. Robis: Robust binary segmentation for high-resolution industrial images. *arXiv preprint arXiv:2505.21152*, 2025. 2, 6

[7] Wei Luo, Yunkang Cao, Haiming Yao, Xiaotian Zhang, Jianan Lou, Yuqi Cheng, Weiming Shen, and Wenyong Yu. Exploring intrinsic normal prototypes within a single image for universal anomaly detection. In *Proceedings of the*

Computer Vision and Pattern Recognition Conference, pages 9974–9983, 2025. 2

[8] Huaiyuan Zhang, Hang Chen, Yu Cheng, Shunyi Wu, Linghao Sun, Linao Han, Zeyu Shi, and Lei Qi. Superad: A training-free anomaly classification and segmentation method for cvpr 2025 vand 3.0 workshop challenge track 1: Adapt & detect. *arXiv preprint arXiv:2505.19750*, 2025. 2, 3, 4, 6, 7

[9] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14318–14328, 2022. 2, 6

[10] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 2

[11] Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025. 2

[12] Kilian Batzner, Lars Heckler, and Rebecca König. Efficientad: Accurate visual anomaly detection at millisecond-level latencies. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 128–138, 2024. 6

[13] Jie Wang, Yanming Zhang, Ting Wang, Yunlong Li, and Jing Chen. An ensemble method for industrial anomaly detection and localization. Technical Report ASEG, GitHub, 2023. Accessed: 2026-05-10. 6

[14] Jianlong Hu, Xu Chen, Zhenye Gan, Jinlong Peng, Shengchuan Zhang, Jiangning Zhang, Yabiao Wang, Chengjie Wang, Liujuan Cao, and Rongrong Ji. Dmad: Dual memory bank for real-world anomaly detection. *arXiv preprint arXiv:2403.12362*, 2024. 6

[15] Ruyi Xu, Yen-Tzu Chiu, Tai-I Chen, Oscar Chew, Yung-Yu Chuang, and Wen-Huang Cheng. Training-free industrial defect generation with diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 24214–24223, 2025. 6