MultiADS: Defect-aware Supervision for Multi-type Anomaly Detection and Segmentation in Zero-Shot Learning

Ylli Sadikaj[*] [1 5 §], Hongkuan Zhou[* 2 3], Lavdim Halilaj[2],
Stefan Schmid[2], Steffen Staab[3 4], Claudia Plant[1 6]

[1]Faculty of Computer Science, University of Vienna, Vienna, Austria
[2]Bosch Corporate Research, Robert Bosch GmbH, Renningen, Germany
[3]University of Stuttgart, Stuttgart, Germany, [4]University of Southampton, Southampton, UK
[5]UniVie Doctoral School Computer Science, University of Vienna, Vienna, Austria
[6]ds:UniVie, Vienna, Austria
{ylli.sadikaj, claudia.plant}@univie.ac.at, steffen.staab@ki.uni-stuttgart.de
{hongkuan.zhou, lavdim.halilaj, stefan.schmid5}@de.bosch.com

Abstract

*Precise optical inspection in industrial applications is crucial for minimizing scrap rates and reducing the associated costs. Besides merely detecting if a product is anomalous or not, it is crucial to know the distinct types of defects, such as a bent, cut, or scratch. The ability to recognize the "exact" defect type enables automated treatments of the anomalies in modern production lines. Current methods are limited to solely detecting whether a product is defective or not, without providing any insights into the defect type, but nevertheless detecting and identifying multiple defects. We propose MultiADS, a zero-shot learning approach, able to perform **Multi**-type **A**nomaly **D**etection and **S**egmentation. The architecture of MultiADS comprises CLIP and extra linear layers to align the visual and textual representation in a joint feature space. To the best of our knowledge, our proposal is the first approach to perform a multi-type anomaly segmentation task in zero-shot learning. Contrary to the other baselines, our approach i) generates specific anomaly masks for each distinct defect type, ii) learns to distinguish defect types, and iii) simultaneously identifies multiple defect types present in an anomalous product. Additionally, our approach outperforms zero/few-shot learning SoTA methods on image-level and pixel-level anomaly detection and segmentation tasks on five commonly used datasets: MVTec-AD, Visa, MPDD, MAD, and Real-IAD.*

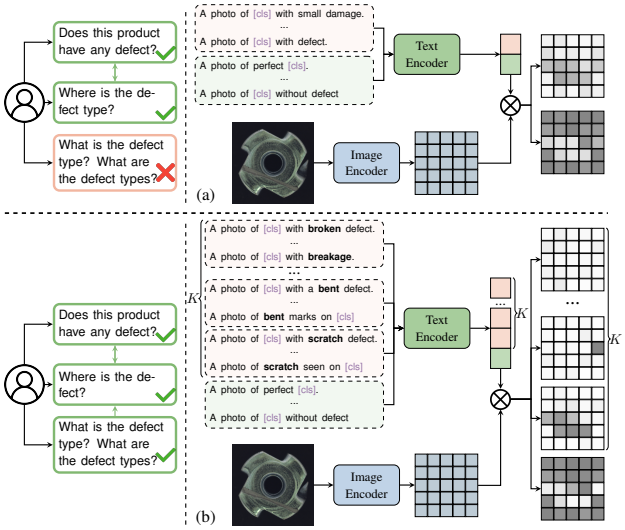
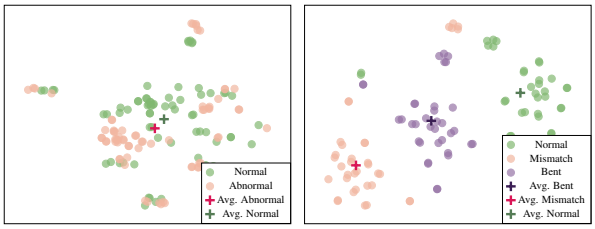


Figure 1. Comparison of common approaches and our approach: a) Common approaches typically differentiate only between normal and abnormal states; whereas b) our approach identifies $K+1$ states: one normal state and K distinct abnormal states corresponding to different defect types. This allows our method to distinguish between various defect types.

1. Introduction

One of the primary objectives of the manufacturing industries is to utilize their assembly lines for a wide range

[*]Both authors contributed equally to this work.
[§]Work done during PhD Sabbatical at Bosch Corporate Research.



(a) Common Approaches (b) Our Approach

Figure 2. Visualization of text prompts (TP) embeddings of common approaches and ours for Bracket Brown product of the MPDD dataset utilizing visualization tool t-SNE [36]. Dot signs $(\cdot)$ represent TP embeddings, plus signs $(+)$ represent the average embedding of TPs with the same color.

of product types. Modern factories are equipped with sophisticated and adaptable mechanisms allowing for a quick reconfiguration to various scenarios [20]. By doing so, the probability of outputting defective products is significantly increased. Therefore, to achieve intelligent manufacturing and prevent downtimes, rework, or quality losses, it is essential to detect anomalies promptly and with high precision [18, 32]. More concretely, identifying the specific defect* type in a product helps operators to understand the underlying causes and effectively implement preventive measures. In this regard, optical inspection via visual anomaly detection and segmentation is crucial to identify abnormal products and locate anomalous regions.

Recent approaches utilize prior knowledge in pre-trained models like CLIP [28] or DINO [4] to boost the generalization performance across a wide range of products for anomaly detection. CLIP-based approaches, such as [5, 16, 44], employ CLIP knowledge and adapt it for anomaly detection and segmentation by defining text-prompts for normal and abnormal states (cf. Figure 1a). Next, they compare the similarity between the image embedding and the average text embedding from generic sets of good and bad prompts. Thus, they are not exploiting anomaly-relevant knowledge, such as defect types, embedded in pre-trained vision language models (VLMs). On the other hand, fine-tuning in the specific domain often leads to overfitting on the training dataset [40], causing the model to lose valuable knowledge critical for accurate anomaly detection and segmentation. In Figure 2a, we visualize how averaging normal and abnormal text embeddings can lead to significant information loss.

In this paper, we present MultiADS, a zero-shot learning approach for multi-type anomaly detection and segmentation that leverages the prior knowledge of the common defect types in VLMs. It aligns the image embedding and the mean text embedding from a general set of good prompts and defect-specific sets of bad prompts. As illustrated in Figure 1b, through our approach, we can answer correctly all three questions, including the question regarding the defect type. Figure 2b shows that MultiADS preserves the meaningful semantic representation within the latent space and clearly distinguishes normal state and distinct defect types. Contrarily, competitive baselines could fail to separate between normal and abnormal states, as shown in Figure 2a. We conduct experiments on five datasets for anomaly detection and anomaly classification, MVTec [1], VisA [46], MPDD [17], MAD (real and simulated) [43], and Real-IAD [37]. We conducted evaluations in both zero-shot/few-shot settings. The empirical results demonstrate that incorporating defect-type information into the learning pipeline improves anomaly detection and segmentation performance across these five datasets. We summarize the key contributions as follows:

- Our MultiADS detects multiple defects of the same and/or different types in an anomalous product. Thus, we propose a new task, namely a multi-type anomaly detection and segmentation task, that aims to determine the defect type at the pixel level. We position MultiADS as a baseline in such a new task.
- We show that by leveraging anomaly-specific knowledge in pre-trained VLMs, MultiADS further improves its detection and segmentation performance.
- We present a Knowledge Base for Anomalies (KBA), that enhances the description of defect types. It can be utilized for defect-aware text prompt construction and facilitates the fine-tuning process of VLMs for anomaly detection and segmentation.
- Additionally, we evaluate the performance of MultiADS on anomaly detection and segmentation against 12 baselines both zero-shot/few-shot settings. The code implementation is publicly available at: https://github.com/boschresearch/MultiADS.

2. Related Work

In this section, we review the most relevant literature based on their learning paradigms and highlight how our approach distinguishes itself from existing methods.

Unsupervised Anomaly Detection. There exists a wide variation in the characteristics of objects and their defects, including differences in color, texture, size, and shape. This heterogeneity leads to an extensive range of defect types, making it challenging to compile a representative set of anomaly samples for training data. Thus, unsupervised anomaly detection approaches, such as [2, 14, 29, 39], require only normal images for train-

*We use *defect* and *anomaly* terms interchangeably.

ing. These methods typically model images without anomalies and classify any deviations from the learned representation as anomalies.

Zero-Shot Anomaly Detection (ZSAD). Recent studies have leveraged the power of large-scale VLMs such as CLIP [28] to perform anomaly detection without any target-specific training. The success of prompt learning in natural language processing has inspired methods such as CoOp [42] and CoCoOP [41], which automatically learn task-specific prompt contexts from only a few labeled examples. Early methods such as WinCLIP [16] and April-GAN [5] adapt CLIP by designing text prompts that differentiate "normal" from "abnormal" states. Also, they introduce window-based strategies or additional linear layers to enhance image segmentation performance.

Other approaches apply the same differentiation technique while adapting the construction for text prompt states. Thus, AnomalyCLIP [44] learns object-agnostic text prompts to capture generic cues of abnormality, SimCLIP [8] further adopts implicit prompt tuning. Similarly, FiLo [11] and AdaCLIP [3] enhance localization by replacing generic anomaly descriptions with adaptively learned fine-grained prompts or tuning hybrid learnable prompts by combining static and dynamic prompts. Contrary to other models, ClipSAM [22] proposes a novel collaboration between CLIP and SAM [19], whereas MuSc [24] detects anomalies by exploiting mutual scoring across unlabeled test images.

Few-Shot Anomaly Detection (FSAD). FSAD models, such as [13, 30, 31, 33], include several normal sample images from the target domain to train their model. PromptAD [25] refines the image–text alignment process by concatenating normal prompts with anomaly-specific suffixes. GraphCore [38] employs graph neural networks to capture rotation-invariant features from limited normal samples, while KAGprompt [34] constructs a kernel-aware hierarchical graph among multi-layer visual features. Other methods adopt reconstruction or feature-matching strategies—such as FastRecon [9] and FOCT [35]-to reconstruct normal appearances from a limited set of normal samples. Given the scarcity of anomalous samples, Anomalydiffusion [12] proposes to employ a latent diffusion model along with spatial anomaly embeddings to generate authentic anomaly image–mask pairs. Meanwhile, AnomalyGPT [10] is an interactive method integrating VLMs to provide defect-specific descriptions for a context-aware inspection. AnomalyDINO [6] uses DINOv2 [27] to extract robust patch-level features for FSAD.

A major limitation of existing vision-language ZSAD and FSAD methods is their binary focus—only distinguishing between normal and abnormal states, as illustrated in Figures 1 and 2. In contrast, MultiADS is designed to perform multi-type anomaly segmentation by constructing defect-specific text prompts that capture rich semantic attributes. This allows MultiADS to not only detect whether an image is anomalous but also to segment and classify the specific type of defect present - a capability that is critical for automated optical inspection in industrial applications.

3. Preliminaries

Here, we introduce the preliminary definitions of binary and multi-type anomaly detection and segmentation, as well as the backbone model.

3.1. Binary Detection and Segmentation

Let $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{target}}$ denote two different datasets, training and target datasets, respectively. Both datasets consist of X, Y, where $X = \{\mathbf{x}_i\}_{i=1}^{N}$ with N images, and $Y = \{(\mathbf{M}_i, y_i)\}_{i=1}^{N}$ with ground truth labels. Each image $\mathbf{x}_i \in \mathbb{R}^{H\times W}$ is masked with $\mathbf{M}_i$ and labeled with y_i, where $y_i \in \{0, 1\}$ is the indicator for anomaly or not and $\mathbf{M}_i \in \{0, 1\}^{H\times W}$ represents the binary anomaly map. Binary anomaly detection and segmentation (BADS) aim to determine if the given image $\mathbf{x}$ contains anomalies and also locate regions in an image that contain anomalies.

3.2. Multi-type Anomaly Segmentation

$\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{target}}$ denote the training and target datasets, respectively. Both datasets consist of X, Y', where $X = \{\mathbf{x}_i\}_{i=1}^{N}$ with N images and $Y' = \{\mathbf{M}'_i\}_{i=1}^{N}$. Each image $\mathbf{x}_i$ is labeled with $\mathbf{M}'_i \in \{0, 1, .., K\}^{H\times W}$, representing the multi-defect segmentation map for one normal class and K abnormal classes. Multi-type anomaly segmentation (MTAS) aims to locate the anomalies and identify various anomaly types.

3.3. Backbone Model

Contrastive Language Image Pre-training (CLIP) is a large-scale vision-language model pre-trained on million-scale image-text pairs, $\{(x_i, t_i)\}_{i=1}^{N}$. It encompasses an image feature encoder, $f(\cdot)$, and a text feature encoder, $g(\cdot)$. CLIP aims to maximize the correlation between $f(x_i)$ and $g(t_i)$ utilizing cosine similarity. Thus, for a given image input x and a closed set of text $T = \{t_1, \ldots, t_K\}$, representing the text prompt for K classes, CLIP performs classification as follows:

$$p(y = j|x) := \frac{exp(\langle f(x), g(t_j)\rangle/\tau)}{\sum_{i=1}^{K} exp(\langle f(x), g(t_i)\rangle/\tau)}, \quad (1)$$

where $\tau > 0$ is the temperature hyperparameter, whereas $\langle \cdot, \cdot \rangle$ represents the cosine similarity.

4. MultiADS Approach

Our proposed approach is a CLIP-based model adapted for zero-shot and few-shot learning for detecting anomalies and identifying the defect types in images from the manufacturing domain. It learns the alignment of image features with their corresponding text features that represent a distinct defect type, as shown in Figures 1 and 3. Anomaly maps constructed for each distinct defect type enable multi-class defect detection and segmentation.

Knowledge Base for Anomalies. We leverage the meta-data from established industrial defect detection datasets, including MVTec-AD, VisA, MPDD, MAD (real and simulated), and Real-IAD, to acquire comprehensive defect-aware information for each product class. Additionally, we incorporate supplementary defect-type properties (attributes) into our knowledge base for anomalies (KBA), including size and shape.

Initially, we group the defect types into superclasses, such that *bent*, *bent lead*, and *bent wire* are represented by the *bent* superclass, similarly *scratch*, *scratch head*, and *scratch neck* are under *scratch*. Thus, we have abstract classes like *bent*, *cut*, *scratch*, capturing all possible defect types that can occur in a given dataset. Details of the acquired information for all datasets part of our KBA are given in the Appendix.

Defect-aware Text Prompts. Next, we utilize the constructed KBA as prior knowledge for our text-prompt construction, as illustrated in Figures 1b and Figure 3. We select the same set of variations of text samples as in [5, 16] to construct text prompts for each given defect class. Figure 2 shows the difference between other baselines and our approach regarding the text prompt embeddings. More details for defect-aware text prompts are provided in the Appendix.

4.1. Training Phase

An overview of the training phase of our proposed method is shown in Figure 3 (LHS). We use different datasets for training and testing with their respective prompt set numbers denoted by K_1 and K_2.

4.1.1. Image and Text Embedding

Each image $\mathbf{x}$ is provided as input to the image encoder to get image patch embeddings at m different stages during encoding, as in [5, 44], $\mathbf{E}_i^p \in \mathbb{R}^{h \times w \times N_i}, i \in \{0, 1, ..., m\}$ with the resolution $h \times w$ and layer N_i, as well as one global image embedding $\mathbf{z}^x \in \mathbb{R}^{N_z}$. We use $K_1 + 1$ sets of text prompts: one representing the normal state and K_1 representing abnormal states corresponding to K_1 defect types. Each set of text prompts is fed into the CLIP text encoder, and we obtain an averaged text embedding for each set by averaging the embeddings of individual prompts. This process yields $K_1 + 1$ averaged text embeddings $\mathbf{z}^t \in \mathbb{R}^{N_z}$, each representing a distinct state.

4.1.2. Aligning Image Patches and Text Prompts

The visual encoder of CLIP is originally trained to align the global object embeddings with text embeddings. To align the two embedding spaces, visual - extracted by the CLIP image encoder, and textual - extracted by the CLIP text encoder, we utilize adapters consisting of a single linear learnable layer. For image patch embeddings at each stage i, a linear adapter takes $\mathbf{E}_i^p$ as input and outputs $\mathbf{Z}_i^p \in \mathbb{R}^{h \times w \times N_z}$. They are compared with $K_1 + 1$ text embeddings $\mathbf{z}^t$ to get the similarity map. Since we choose image patches embeddings at m different stages, we get m similarity maps $\mathbf{S}_i \in \mathbb{R}^{(K_1+1) \times h \times w}$, where h, w are the resolution of the similarity maps, K_1 is the number of defect types. Each map $\mathbf{S}_i$ is up-sampled to match the size of the input image and aligned with the ground truth segmentation map $\mathbf{M}'_x$.

4.1.3. Training Objective

Two typical losses, focal [26] and dice [23], are used for segmentation tasks. Focal loss is designed to address class imbalance issues, especially in tasks like object detection, where there is often a significant imbalance between classes. We face the same challenge, i.e., a high number of normal images and a low number of abnormal images; therefore, we apply a multi-class focal loss for multi-defect segmentation along with the binary dice loss for anomaly segmentation. These two training objectives are combined to form the final loss function:

$$\mathcal{L} = \sum_{i=1}^{m} \mathcal{L}_{\text{focal}}(UP(\mathbf{S}_i), \mathbf{M}'_x) + \mathcal{L}_{\text{dice}}(\mathbf{1} - UP(\mathbf{S}_i)[0], \mathbf{M}_x), \quad (2)$$

where $\mathbf{M}'_x$ represents the ground truth multi-defect segmentation map, and $\mathbf{M}_x$ is the binary anomaly map. $UP(\cdot)$ denotes the up-sampling function used to scale the similarity map to the input image resolution. Note that in the training phase, the global anomaly score a_x is not fine-tuned.

4.2. Inference Phase

To test the trained model's performance in the target dataset, we first construct $K_2 + 1$ sets of text prompts, representing one normal state without defect and K_2

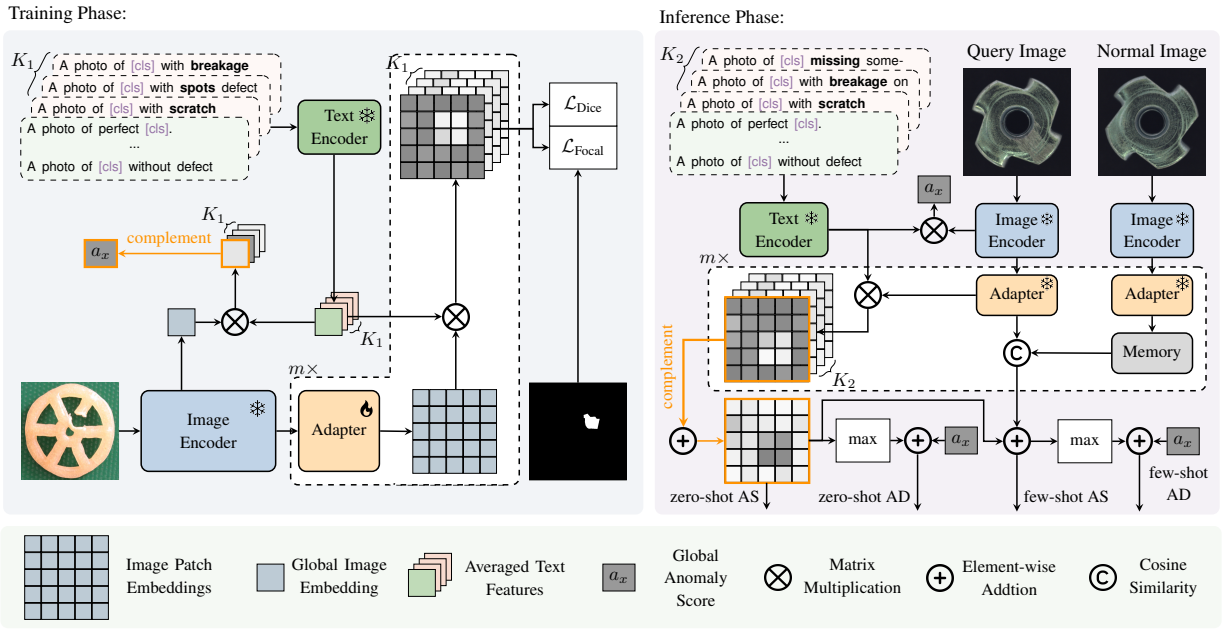


Figure 3. Training phase: K_1 text prompts describing the defect types plus one for good products are encoded into K_1+1 averaged text embeddings. The image patches are encoded and compared to these embeddings to produce K_1+1 similarity maps. For multi-type anomaly segmentation, we use dice and focal loss. Inference phase: we construct K_2+1 sets of text prompts. For anomaly segmentation (AS), we up-sample the complement of the normal layer's similarity map. For anomaly detection (AD), the global anomaly score a_x and the maximum score from the anomaly map are utilized. In few-shot testing, the query image is then compared with multiple reference (normal) images in the testing dataset to generate a similarity map. This similarity map is finally up-sampled and combined with the anomaly map for segmentation and classification tasks.

states representing distinct defect types of the target domain. An overview of the inference phase of our proposed method is shown in Figure 3 (RHS).

Each set of text prompts is input into the CLIP text encoder to generate embeddings, while the query image is passed through the CLIP image encoder and then the adapter to produce m similarity maps $\mathbf{S}_i \in \mathbb{R}^{(K_2+1)\times h\times w}$. The respective similarity maps are then up-sampled to match the original size of the input image. The multi-defect segmentation map is calculated by averaging the up-sampled similarity map:

$$\hat{\mathbf{M}}'_x = \frac{1}{m}\sum_{i=1}^{m} UP(\mathbf{S}_i). \tag{3}$$

We only take the *first* layer of similarity maps and perform a complement operation on each pixel to create the anomaly score map. Since there are m similarity maps, we average the m anomaly score maps to obtain the final anomaly map:

$$\hat{\mathbf{M}}_x = \frac{1}{m}\sum_{i=1}^{m} \mathbf{1} - UP(\mathbf{S}_i)[0]. \tag{4}$$

The global image embedding $\mathbf{z}^x$ from the pre-trained CLIP image encoder is also compared with K_2+1 text embeddings to get K_2+1 global similarity scores. After the normalization, the complement of the similarity score compared to the normal state text prompts is used as the final global anomaly score a_x. We perform zero-shot learning based on the acquired anomaly map $\hat{\mathbf{M}}_x$ and global anomaly score a_x. Few-shot learning is conducted based on the acquired anomaly map $\hat{\mathbf{M}}_x$, global anomaly score a_x, and reference anomaly map $\hat{\mathbf{M}}_{\text{ref}}$ between query image and reference normal image(s).

4.2.1. Multi-type Anomaly Segmentation

The m similarity maps $\boldsymbol{S}_i, i \in \{1,\ldots,m\}$, are up-sampled to match the input image size and then averaged to produce the multi-defect segmentation map, $\hat{\mathbf{M}}'_x \in \mathbb{R}^{(K_2+1)\times h\times w}$. This map captures both the anomaly locations and their respective defect types, enabling effective support for the multi-type anomaly segmentation task.

4.2.2. Zero-shot Learning

For zero-shot learning, the output anomaly map $\hat{\mathbf{M}}_x$ is used for anomaly segmentation and compared with the ground truth labels. The highest anomaly score: $\max(\hat{\mathbf{M}}_x)$ on anomaly map and global anomaly score

a_x are averaged and then compared against a threshold θ to determine whether the image contains an anomaly.

4.2.3. Few-shot Learning

To conduct few-shot learning, we need to compute an extra reference anomaly map based on the similarity between the query image and several reference normal images. The reference normal image(s) are fed into the image encoder to get m stages of image patch embeddings. We leverage memory banks [5] to store the features of the reference images, which can be compared with input image features by cosine similarity to obtain the reference anomaly map $\hat{M}_{\text{ref}}$. The final anomaly map $\hat{\mathbf{M}}_{\text{final}} = \frac{1}{2}(\hat{\mathbf{M}}_x + \hat{\mathbf{M}}_{\text{ref}})$ is used for anomaly segmentation. $\hat{\mathbf{M}}_{\text{final}}$ instead of $\hat{\mathbf{M}}_x$ is used to determine the anomaly itself.

4.2.4. Filtering Out Product-irrelevant Defect Types

For a specific product type, only certain defect types are relevant. During the inference phase, this filtering step involves excluding text prompt sets associated with defect types that are not applicable to the product, ensuring that only relevant defect types are considered. Here, the method that includes this filtering process is referred to as MultiADS-F, while the original version without filtering remains as MultiADS.

5. Experiments

In this section, we describe datasets and baselines and discuss the results of the conducted experiments.

5.1. Datasets

Five common datasets: MVTec-AD [1], VisA [46], MPDD [17], MAD (simulated and real) [43], and Real-IAD [37] are used for the multi-type anomaly segmentation as well as the binary anomaly detection and segmentation task, respectively. More details of these datasets are provided in the Appendix.

5.2. Experiment Setting

We adopt a transfer learning setting, where the model is trained on one of the datasets and evaluated on the remaining. In the zero-shot learning scenario, the trained model is directly applied to the target dataset without any additional information from the target dataset. In contrast, the few-shot learning scenario allows the trained model to access a small number of normal images from the target dataset for further adaptation.

We use the ViT-L-14-336 CLIP backbone from OpenCLIP [15], pre-trained on the LAION-400M_E32 setting of open-clip. The learning rate is set to 0.001, with a batch size of 8. The stage number $m = 4$. The features are selected from layers: 6, 12, 18, and 24.

5.3. Evaluation Metrics

We assess the anomaly detection performance on zero/few-shot learning settings with three metrics, namely the receiver-operator curve (AUROC), the F1-score at the optimal threshold (F1-max), and the average precision (AP). Similar to [5, 16, 44], the anomaly segmentation is quantified by AUROC, F1-max, and the per-region overlap (PRO) of the segmentation using the pixel-wise anomaly scores. For the multi-type anomaly segmentation task, we employ AUROC, F1-score, and AP with the macro averaging setting.

5.4. Baselines

We compare the performance of our approach with the following 12 baselines: CLIP [28], CLIP-AC [28], CoOp [42], CoCoOp [41], PatchCore [30], WinCLIP [16], April-GAN [5], InCTRL [45], PromptAD [25], AnomalyCLIP [44], AdaCLIP [3], and AnomalyGPT [10]. CLIP, CLIP-AC, CoCo, CoCoOP, WinCLIP, April-GAN, AnomalyCLIP, and AdaCLIP are zero-shot learning approaches. Whereas CoOp, WinCLIP, and April-GAN can also learn in the few-shot setting, as other approaches, PatchCore, PromptAD, InCTRL, and AnomalyGPT. The comparison of batch zero-shot setting with MuSc [24] and AnomalyDINO [6] is discussed in the Appendix. We did not include other baselines such as [8, 11, 22] because their authors did not provide implementation yet.

In the evaluation process, we use the basic approach, MultiADS, and the filtering-based variant, MultiADS-F.

5.5. Results

Next, we present and discuss results from the experiments for multi-type anomaly segmentation in zero-shot settings and binary ZSAD and FSAD.

5.5.1. Multi-type Anomaly Segmentation

First, we discuss our MultiADS's performance in the new task, the multi-type anomaly segmentation (MTAS) task, which can segment various defect types. To the best of our knowledge, we are the first to perform such a task, and thus we present MultiADS as a baseline.

Table 1 shows the results of MultiADS on the MTAS task in a zero-shot learning setting. We observe that our approach achieves high accuracy in terms of the AUROC metric for pixel-level segmentation of distinct defects in all datasets. As expected, MultiADS performs with higher accuracy in terms of AP metric on datasets

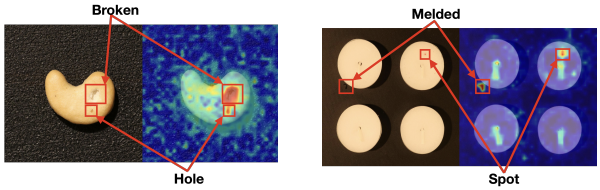
Table 1. Results on MTAS Task of MultiADS.

Train	Target	Pixel-Level		
		AUROC	F1-score	AP
MVTec-AD	VisA	93.6	22.3	24.8
	MPDD	95.2	42.8	53
	MAD-sim	92.1	27.9	31.5
	MAD-real	89.2	52.5	52.3
	Real-IAD	89.5	22.6	25.0
VisA	MVTec-AD	89.1	24	30.5
	MPDD	95.3	46.7	50.5
MPDD	VisA	93.4	22.1	23.3
	MVTec-AD	89.4	23.9	27.6
Real-IAD	MVTec-AD	87.7	21.4	29.9
	VisA	88.1	23.8	24.8

with fewer anomaly types, such as MPDD and MAD-real, and the accuracy is slightly lower on datasets with multiple anomaly types appearing concurrently, such as Real-IAD and VisA. Additionally, we found that MultiADS performs slightly better on the VisA dataset when our model is trained on the MVTec-AD or Real-IAD datasets rather than the MPDD dataset due to higher similarity between defect types of the VisA dataset with MVTec-AD and Real-IAD datasets. Similarly, the VisA dataset serves as a good model trainer regarding the performance of the model on the MVTec-AD dataset. In summary, these results indicate that MultiADS can successfully differentiate between various defect types. We provide more results on the MTAS task in the Appendix.

Multi-type Anomaly Awareness. Figure 4 shows that multiple defect types, such as *broken* and *hole*, can appear on one image, and MultiADS can successfully locate and classify these defects. Additionally, in Table 2, we listed the segmentation performance for some sample defect types that are seen/unseen during the training phase. We notice that defects such as *holes* and *damages* are relatively easy to locate and classify because they also occur on the training dataset - MVTec-AD. It may be that these defects are similar in terms of shape to those they have in datasets. For unseen defects like *extra* and *stuck*, our model achieves slightly lower accuracy. On the other hand, for other unseen defects such as *pit*, we can still perform with high accuracy on the classification task. These results reflect that our approach has generalization ability even on large and complex datasets and unseen defects in the training dataset.

Ablation Study. We present the results of our ablation studies on MTAS, quantifying the contributions of the KBA component. As Table 3 shows, the performance improves with the detailed text prompts constructed by KBA in both VisA and MAD-sim datasets. Similar patterns are present across all datasets.



(a) Broken and Hole defects. (b) Melded and Spot defects.

Figure 4. MultiADS locates and identifies simultaneously multi-type anomalies on cashew (a) and candle (b) products.

Table 2. Results MTAS for zero-shot setting at pixel-level for sample defect-types. The model is trained on the MVTec-AD dataset. - indicates **unseen** defect types while ✓indicates **seen** defect types during training.

(a) VisA

	Defects	AUROC	F1-Score	AP
-	Extra	94.07	2.11	0.15
-	Stuck	91.54	10.51	7.76
✓	Bent	96.53	6.07	7.74
✓	Hole	99.55	12.64	25.19

(b) Real-IAD

	Defects	AUROC	F1-Score	AP
-	Pit	97.08	6.15	1.01
✓	Contamin.	90.03	6.12	1.86
✓	Scratch	92.63	4.37	2.96
✓	Damage	96.61	6.31	9.75

Table 3. Ablation studies on the role of KBA for MTAS

	MVTec → VisA			MVTec → MAD-sim		
KBA	AUROC	F1-score	AP	AUROC	F1-score	AP
-	87.0	22.1	23.6	91.1	25.1	26.5
✓	93.6	22.3	24.8	92.1	27.9	31.5

5.5.2. Binary Detection and Segmentation

ZSAD. In Table 4, we show the performance on ZSAD for pixel-level (AUROC, AUPRO) and image-level (AUROC, AP) on VisA, MPDD, MAD (sim and real), and Real-IAD datasets. We selected these metrics to evaluate the performance following [44]. For a fair comparison, our approach and baseline approaches, including WinCLIP, April-GAN, AnomalyCLIP, and AdaCLIP, are trained on the MVTec-AD dataset. We observe that MultiADS and MultiADS-F are the best overall performers, especially when performance is evaluated with the AUPRO and AUROC metrics at the pixel and image levels, respectively. We note that our approach achieves the best performance for all metrics on both levels on the recent datasets, MAD and Real-IAD, which are even more challenging. Meanwhile, MultiADS-F is the best overall performer on the MPDD, MAD-real, and Real-IAD datasets, indicating that text prompts of non-relevant defect types present more noise for these datasets. Note that MultiADS and MultiADS-F have the same scores for the MAD-sim dataset, as all defect types appear for all product types. The best baseline performer is the AnomalyCLIP approach.

Table 5 shows the ablation study quantifying the contributions of KBA, global anomaly score, and different stage numbers on the ZASD task. The stage number has

Table 4. Zero-shot anomaly detection and segmentation. (Bold represents best performer; underline indicates second best performer, * means results are taken from papers)

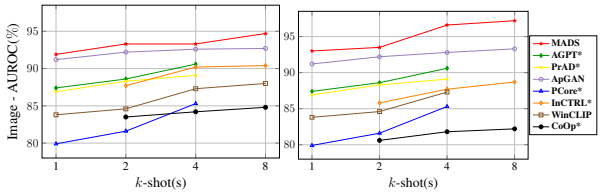
ZSAD			Pixel-Level		Image-Level	
Dataset	Method	Venue	AUROC	AUPRO	AUROC	AP
VisA	CLIP*	ICML21	46.6	14.8	66.4	71.5
	CLIP-AC*	ICML21	47.8	17.3	65.0	70.1
	CoOp*	IJCV22	24.2	3.8	62.8	68.1
	CoCoOp*	CVPR22	93.6	-	78.1	-
	WinCLIP	CVPR23	79.6	56.8	78.1	81.2
	April-GAN	CVPR23	94.2	86.8	78.0	81.4
	AnomalyCLIP	CVPR24	**95.5**	87.0	82.1	85.4
	AdaCLIP	ECCV24	95	-	75.4	79.3
	MultiADS (ours)		95	**89.7**	**83.6**	**86.9**
	MultiADS-F (ours)		94.5	87.4	82.5	86.5
MPDD	CLIP*	ICML21	62.1	33.0	54.3	65.4
	CLIP-AC*	ICML21	58.7	29.1	56.2	66.0
	CoOp*	IJCV22	15.4	2.3	55.1	64.2
	CoCoOp*	CVPR22	95.2	-	61	-
	WinCLIP	CVPR23	76.4	48.9	63.6	69.9
	April-GAN	CVPR23	94.1	83.2	73.0	80.2
	AnomalyCLIP	CVPR24	**96.5**	88.7	77.0	**82.0**
	AdaCLIP	ECCV24	96.3		66.3	75
	MultiADS (ours)		95.8	**89.7**	78.3	78.4
	MultiADS-F (ours)		96.3	89.5	**79.7**	80.5
MAD-sim	WinCLIP	CVPR23	77.6	55.8	54.3	90.2
	April-GAN	CVPR23	80.4	61.5	56	91
	AnomalyCLIP	CVPR24	77.9	40.1	54.6	90.9
	AdaCLIP	ECCV24	85.7	-	55.2	90.5
	MultiADS (ours) MultiADS-F (ours)		**88.0**	**74.2**	**57.1**	**94.4**
MAD-real	WinCLIP	CVPR23	60.5	26.9	64.1	87.6
	April-GAN	CVPR23	88.2	69.5	62.9	87.7
	AnomalyCLIP	CVPR24	88.3	65.1	66.8	90
	AdaCLIP	ECCV24	85.7	-	59	86.5
	MultiADS (ours)		89.7	74.0	78.3	**92.9**
	MultiADS-F (ours)		**90.7**	**75.2**	**78.5**	**92.9**
Real-IAD	WinCLIP	CVPR23	87.1	59.9	75	72.3
	April-GAN	CVPR23	96	86.8	75.7	73.5
	AnomalyCLIP	CVPR24	96.2	85.7	78.4	76.7
	AdaCLIP	ECCV24	95.3	-	70.1	68.5
	MultiADS (ours)		**96.6**	87.1	**78.7**	**79.1**
	MultiADS-F (ours)		96.3	**87.2**	78.2	78.5

the highest impact; the drop in performance is around 5% in terms of AP for both datasets when $m = 3$.

Table 5. Ablation studies on the role of KBA, global anomaly score a_x, and stage number m on the ZSAD task. Pixel-level results are ignored since a_x is only used at the image-level.

ZSAD			MVTec → VisA				MVTec → MPDD			
			Pixel-Level		Image-Level		Pixel-Level		Image-Level	
m	a_x	KBA	AUROC	AUPRO	AUROC	AP	AUROC	AUPRO	AUROC	AP
3	✓	✓	94.5	87.7	79.5	82.3	93.7	84.3	68.2	74.8
4	-	✓	-	-	82.1	85.8	-	-	76.5	78.1
4	✓	-	94.4	88.7	82.4	86.1	95.7	89.1	77.9	77.6
4	✓	✓	95.0	89.7	83.6	86.9	95.8	89.5	78.3	78.4

FSAD. Figure 5 shows the results for the FSAD task, for image-level (AUROC) with different numbers of shots, $k = [1, 2, 4, 8]$, on the Visa and MVTec-AD datasets. Similarly to ZSAD, we train our model on the MVTec-AD dataset and test on VisA and vice versa. We note that the most competitive baselines are April-GAN, PromptAD, and AnomalyGPT. We observe that MultiADS is the best overall performer for both datasets. The same performance patterns are found on other datasets,



(a) VisA dataset (b) MVTec-AD dataset

Figure 5. Few-Shot Image level (AUROC) accuracy for different k-shots on the VisA and MVTec-AD datasets. (* - results taken from papers, AGPT - AnomalyGPT, PCore - PatchCore, PrAD - PromptAD, ApGAN - April-GAN)

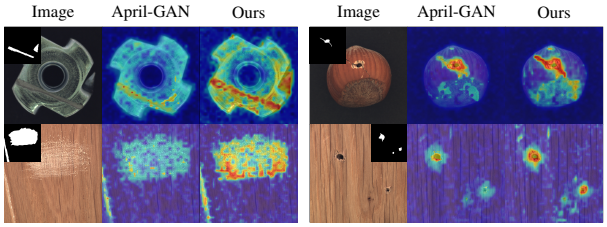


Figure 6. Visualization of anomaly segmentation from VisA and MVTec-AD datasets, in the few-shot (k=4) for defect types - scratch and hole. Each anomaly is highlighted to illustrate the ability of April-GAN and MultiADS.

too. The main advantage of our approach lies in extending the investigation based on defect awareness, supporting our claim that the main drawback of other methods is the two-state (normal and abnormal) limitation.

Figure 6 depicts a qualitative evaluation of the FSAD results of MultiADS and the best overall competitor, April-GAN, for scratch and hole defect types. We observe that MultiADS demonstrates higher confidence in identifying anomalies and achieves better segmentation across the same and different defect types due to its enhanced ability to capture the semantics of different defect types. More results are provided in the Appendix.

6. Conclusion

In this paper, we propose MultiADS, which constructs defect-aware text prompts to improve the performance of anomaly detection and segmentation tasks. We present a multi-type anomaly segmentation task that aims to determine the defect types and locations at the pixel level. We evaluated MultiADS on such a new task and positioned it as a baseline that can be used by the community. Finally, we evaluate MultiADS's performance against 12 baselines in ZSAD/FSAD on five datasets. Our evaluation demonstrates that MultiADS achieves the best performance in most cases for ZSAD/FSAD. In the future, we plan to explore adapting our approach to learn text prompt embeddings.

7. Acknowledgement

This work was partially funded by the European Union's Horizon RIA research and innovation programme under grant agreement No. 101092908 (SMARTEDGE). The authors also thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Hongkuan Zhou.

References

[1] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad — a comprehensive real-world dataset for unsupervised anomaly detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9584–9592, 2019. 2, 6, 12, 13, 24

[2] Paul Bergmann, Kilian Batzner, Michael Fauser, David Sattlegger, and Carsten Steger. Beyond dents and scratches: Logical constraints in unsupervised anomaly detection and localization. *International Journal of Computer Vision*, 130(4):947–969, 2022. 2

[3] Yunkang Cao, Jiangning Zhang, Luca Frittoli, Yuqi Cheng, Weiming Shen, and Giacomo Boracchi. Adaclip: Adapting clip with hybrid learnable prompts for zero-shot anomaly detection. In *European Conference on Computer Vision*, 2024. 3, 6

[4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, pages 9630–9640. IEEE, 2021. 2

[5] Xuhai Chen, Yue Han, and Jiangning Zhang. A zero-/few-shot anomaly classification and segmentation method for cvpr 2023 vand workshop challenge tracks 1&2: 1st place on zero-shot ad and 4th place on few-shot ad. *arXiv preprint arXiv:2305.17382*, 2023. 2, 3, 4, 6, 12, 21

[6] Simon Damm, Mike Laszkiewicz, Johannes Lederer, and Asja Fischer. Anomalydino: Boosting patch-based few-shot anomaly detection with dinov2. *CoRR*, abs/2405.14529, 2024. 3, 6, 24

[7] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. Padim: A patch distribution modeling framework for anomaly detection and localization. In *Pattern Recognition. ICPR International Workshops and Challenges*, pages 475–489, Cham, 2021. Springer International Publishing. 21

[8] Chenghao Deng, Haote Xu, Xiaolu Chen, Haodi Xu, Xiaotong Tu, Xinghao Ding, and Yue Huang. Simclip: Refining image-text alignment with simple prompts for zero-/few-shot anomaly detection. In *Proceedings of the 32nd ACM International Conference on Multimedia*, page 1761–1770, New York, NY, USA, 2024. Association for Computing Machinery. 3, 6

[9] Zheng Fang, Xiaoyang Wang, Haocheng Li, Jiejie Liu, Qiugui Hu, and Jimin Xiao. Fastrecon: Few-shot industrial anomaly detection via fast feature reconstruction. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 17435–17444, 2023. 3, 22

[10] Zhaopeng Gu, Bingke Zhu, Guibo Zhu, Yingying Chen, Ming Tang, and Jinqiao Wang. Anomalygpt: Detecting industrial anomalies using large vision-language models. *arXiv preprint arXiv:2308.15366*, 2023. 3, 6

[11] Zhaopeng Gu, Bingke Zhu, Guibo Zhu, Yingying Chen, Hao Li, Ming Tang, and Jinqiao Wang. Filo: Zero-shot anomaly detection by fine-grained description and high-quality localization. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 2041–2049, 2024. 3, 6

[12] Teng Hu, Jiangning Zhang, Ran Yi, Yuzhen Du, Xu Chen, Liang Liu, Yabiao Wang, and Chengjie Wang. Anomalydiffusion: Few-shot anomaly image generation with diffusion model. In *Proceedings of the AAAI conference on artificial intelligence*, pages 8526–8534, 2024. 3

[13] Chaoqin Huang, Haoyan Guan, Aofan Jiang, Ya Zhang, Michael Spratling, and Yan-Feng Wang. Registration based few-shot anomaly detection. In *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIV*, page 303–319, Berlin, Heidelberg, 2022. Springer-Verlag. 3, 21

[14] Chaoqin Huang, Haoyan Guan, Aofan Jiang, Ya Zhang, Michael Spratling, and Yan-Feng Wang. Registration based few-shot anomaly detection. In *Computer Vision – ECCV 2022*, pages 303–319, Cham, 2022. Springer Nature Switzerland. 2

[15] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, 2021. 6, 22

[16] Jongheon Jeong, Yang Zou, Taewan Kim, Dongqing Zhang, Avinash Ravichandran, and Onkar Dabeer. Winclip: Zero-/few-shot anomaly classification and segmentation. In *CVPR*, pages 19606–19616. IEEE, 2023. 2, 3, 4, 6, 21

[17] Stepan Jezek, Martin Jonak, Radim Burget, Pavel Dvorak, and Milos Skotak. Deep learning-based defect detection of metal parts: evaluating current methods in complex conditions. In *2021 13th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pages 66–71, 2021. 2, 6, 12, 13, 24

[18] Yu Jiang, Wei Wang, and Chunhui Zhao. A machine vision-based realtime anomaly detection method for industrial products using deep learning. In *2019 Chinese Automation Congress (CAC)*, pages 4842–4847, 2019. 2

[19] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo,

et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023. 3

[20] Tomas Kliestik, Marek Nagy, and Katarina Valaskova. Global value chains and industry 4.0 in the context of lean workplaces for enhancing company performance and its comprehension via the digital readiness and expertise of workforce in the v4 nations. *Mathematics*, 11 (3), 2023. 2

[21] Aodong Li, Chen Qiu, Marius Kloft, Padhraic Smyth, Maja Rudolph, and Stephan Mandt. Zero-shot anomaly detection via batch normalization. In *NeurIPS*, 2023. 24

[22] Shengze Li, Jianjian Cao, Peng Ye, Yuhan Ding, Chongjun Tu, and Tao Chen. Clipsam: Clip and sam collaboration for zero-shot anomaly segmentation. *Neurocomputing*, 618:129122, 2025. 3, 6

[23] Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. Dice loss for data-imbalanced NLP tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 465–476. Association for Computational Linguistics, 2020. 4

[24] Xurui Li, Ziming Huang, Feng Xue, and Yu Zhou. Musc: Zero-shot industrial anomaly classification and segmentation with mutual scoring of the unlabeled images. In *International Conference on Learning Representations*, 2024. 3, 6, 24

[25] Xiaofan Li, Zhizhong Zhang, Xin Tan, Chengwei Chen, Yanyun Qu, Yuan Xie, and Lizhuang Ma. Promptad: Learning prompts with only normal samples for few-shot anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16838–16848, 2024. 3, 6, 22

[26] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007, 2017. 4

[27] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision. *Trans. Mach. Learn. Res.*, 2024, 2024. 3

[28] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763. PMLR, 2021. 2, 3, 6, 13, 21

[29] 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 (CVPR)*, 2022. 2

[30] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14298–14308, 2022. 3, 6, 21

[31] Marco Rudolph, Bastian Wandt, and Bodo Rosenhahn. Same same but differnet: Semi-supervised defect detection with normalizing flows. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1906–1915, 2021. 3

[32] Patrick Ruediger-Flore, Matthias Klar, Marco Hussong, Avik Mukherjee, Moritz Glatt, and Jan C. Aurich. Comparing binary classification and autoencoders for vision-based anomaly detection in material flow. *Procedia CIRP*, 121:138–143, 2024. 2

[33] Shelly Sheynin, Sagie Benaim, and Lior Wolf. A hierarchical transformation-discriminating generative model for few shot anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8495–8504, 2021. 3

[34] Fenfang Tao, Guo-Sen Xie, Fang Zhao, and Xiangbo Shu. Kernel-aware graph prompt learning for few-shot anomaly detection, 2025. 3

[35] Long Tian, Hongyi Zhao, Ruiying Lu, Rongrong Wang, Yujie Wu, Liming Wang, Xiongpeng He, and Xiyang Liu. Foct: Few-shot industrial anomaly detection with foreground-aware online conditional transport. In *Proceedings of the 32nd ACM International Conference on Multimedia*, page 6241–6249, New York, NY, USA, 2024. Association for Computing Machinery. 3

[36] L.J.P. van der Maaten and G.E. Hinton. Visualizing high-dimensional data using t-sne. *JMLR*, 9:2579–2605, 2008. 2

[37] Chengjie Wang, Wenbing Zhu, Bin-Bin Gao, Zhenye Gan, Jiangning Zhang, Zhihao Gu, Shuguang Qian, Mingang Chen, and Lizhuang Ma. Real-iad: A real-world multi-view dataset for benchmarking versatile industrial anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22883–22892, 2024. 2, 6, 12, 13, 24

[38] Guoyang Xie, Jinbao Wang, Jiaqi Liu, Yaochu Jin, and Feng Zheng. Pushing the limits of fewshot anomaly detection in industry vision: Graphcore. In *The Eleventh International Conference on Learning Representations*, 2023. 3, 21

[39] Guoyang Xie, Jinbao Wang, Jiaqi Liu, Feng Zheng, and Yaochu Jin. Pushing the limits of fewshot anomaly detection in industry vision: Graphcore, 2023. 2

[40] Hongkuan Zhou, Lavdim Halilaj, Sebastian Monka, Stefan Schmid, Yuqicheng Zhu, Bo Xiong, and Steffen Staab. Visual representation learning guided by multimodal prior knowledge. *CoRR*, abs/2410.15981, 2024. 2

[41] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition(CVPR)*, 2022. 3, 6, 21

[42] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision (IJCV)*, 2022. 3, 6, 21

[43] Qiang Zhou, Weize Li, Lihan Jiang, Guoliang Wang, Guyue Zhou, Shanghang Zhang, and Hao Zhao. Pad: A dataset and benchmark for pose-agnostic anomaly detection. *arXiv preprint arXiv:2310.07716*, 2023. 2, 6, 12, 13, 24

[44] Qihang Zhou, Guansong Pang, Yu Tian, Shibo He, and Jiming Chen. Anomalyclip: Object-agnostic prompt learning for zero-shot anomaly detection. In *ICLR*. OpenReview.net, 2024. 2, 3, 4, 6, 7, 21, 22

[45] Jiawen Zhu and Guansong Pang. Toward generalist anomaly detection via in-context residual learning with few-shot sample prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17826–17836, 2024. 6, 21, 22

[46] Yang Zou, Jongheon Jeong, Latha Pemula, Dongqing Zhang, and Onkar Dabeer. Spot-the-difference self-supervised pre-training for anomaly detection and segmentation. *arXiv preprint arXiv:2207.14315*, 2022. 2, 6, 12, 13, 24

MultiADS: Defect-aware Supervision for Multi-type Anomaly Detection and Segmentation in Zero-Shot Learning

Supplementary Material

8. Our approach

In this section, we will further discuss more details regarding our proposed approach, MultiADS.

8.1. Knowledge Base for Anomalies and Defect-Aware Text Prompts Design

We construct text prompts based on the information we obtain from the Knowledge Base for Anomalies (KBA). This allows for leveraging the specificity of the defect type for each product class. The procedure for defect-aware prompt construction is consistently applied to each dataset. It should be noted, however, that the text prompt regarding the normal state and text template are the same for all datasets.

We conduct experiments on three commonly known datasets, namely MVTec-AD [1], VisA [46], MPDD [17], MAD [43], Real-IAD [37]. We construct multiple distinct defect-aware text prompts and 1 for the normal state, for each dataset. We construct text prompts that represent the normal or good state (without defects) of the images, using the following text prompt template:

normal = ["[cls]", "flawless [cls]", "perfect [cls]", "unblemished [cls]", "[cls] without flaw", "[cls] without defect", "[cls] without damage", "[cls] with immaculate quality", "[cls] without any imperfections", '[cls] in ideal condition"]

where *[cls]* represents a product class from a given dataset. We apply the same normal state design for all datasets, utilizing the text template as in [5] for all datasets as follows:

text-template = ["a bad photo of a {}.","a low resolution photo of the {}.", "a bad photo of the {}.", "a cropped photo of the {}.", "a bright photo of a {}.", "a dark photo of the {}.", "a photo of my {}.", "a photo of the cool {}.", "a close-up photo of a {}.", "a black and white photo of the {}.", "a bright photo of the {}.", "a cropped photo of a {}.", "a jpeg corrupted photo of a {}.", "a blurry photo of the {}.", "a photo of the {}.", "a good photo of the {}.", "a photo of one {}.", "a close-up photo of the {}.", "a photo of a {}.", "a low resolution photo of a {}.", "a photo of a large {}.", "a blurry photo of a {}.", "a jpeg corrupted photo of the {}.", "a good photo of a {}.", "a photo of the small {}.", "a photo of the large {}.", "a black and white photo of a {}.", "a dark photo of a {}.", "a photo of a cool {}.", "a photo of a small {}.", "this is a {} in the scene.", "this is the {} in the scene.", "this is one {} in the scene.", "there is the {} in the scene.", "there is a {} in the scene."]

where {} is filled with content from the normal and defect-aware text prompts.

An example of a text-prompt representing the normal state for product class *[cls]* = *cable* is as follows:

$$S_{\text{normal}} = \{\text{“A bad photo of } \textit{cable}\text{.”}, \cdots, \text{“There is a } \textit{cable in ideal condition} \text{ in the scene.”}\} \tag{5}$$

Similarly, we construct text prompts representing distinct defect types. An example of a text-prompt representing the *bent* defect type for product class *[cls]* = *cable* is as follows:

$$S_{\text{bent}} = \{\text{“A bad photo of } \textit{cable has a bent defect}\text{.”}, \cdots, \text{“There is a } \textit{bent edge on cable} \text{ in the scene.”}\} \tag{6}$$

In Tables 7-11, we show the defect-aware text prompts for each defect type for all datasets, respectively. Note that for shared defect types among the datasets, such as *bent*, *hole*, and *scratch*, we use the same defect-aware text prompts among all datasets.

We provide the defined defect-aware text prompts, attached to the source code. The simplest way is to adapt the defect-aware information in a suitable manner based on the design of other approaches that aim to investigate defect types in anomaly detection tasks.

In the main manuscript, we mention that the KBA contains the information for defect variations and defect type properties (attributes). Also, we include synonyms of defect types such as *a slight curve*, which can also help VLMs to capture the similarity between image-text pairs. Likewise, we apply the same strategy for the construction of defect-aware text prompts for all defect types. More examples are provided in Tables 7-11. Additionally, Tables 12-17 show variations of each defect

type observed from all given datasets, for example *bent* contains variations *bent lead*, *bent wire*, and *bent edge*.

9. Datasets

Table 6. Key statistics on the datasets.

Dataset	Category	$\|\mathcal{C}\|$	Normal / Anomalous Samples
MVTec-AD [1]	Object Texture	15	4,096 / 1,258
VisA [46]	Object	12	9,621 / 1,200
MPDD [17]	Object	6	1,064 / 282
MAD [43]	Object	20	5,231 / 4,902
Real-IAD [37]	Object	30	99,721 / 51,329

Due to space limitations in the main manuscript, here we describe in detail the industrial anomaly detection datasets: MVTec-AD [1], VisA [46], MPDD [17], MAD (simulated and real) [43], and Real-IAD [37]. Key statistics on the datasets are shown in Table 6, such as categories, distinct classes, and the number of samples. MVTec-AD dataset consists of two categories, namely objects and textures, and 15 product classes. For each product, there can be a different number of defects, as shown in Table 12. This number varies from 1 up to 8, but for the textures, it is 5 for all products. We classify each defect to the defect type as we defined before.

Additionally, we provide more details about defect types in order to highlight the importance and the design of our defect-aware text prompts. Thus, details of the VisA datasets are shown in Table 13; the products are categorized into complex structures, multiple instances (an image with multiple products of the same class, e.g., multiple candles, multiple capsules), and single instances. In total, it consists of 130 defect types if we consider different combinations of defect types, but if we consider the combination as a single defect type, then the VisA dataset has 84 defect types and 40 distinct defect types. In Table 13, some defect types are included as part of the *Combined* defect type, which consists of multiple defect types. The number of defect types for each product varies between 5 and 9 defect types. In Table 14, we show detailed information regarding the MPDD dataset, which consists of 6 product types and 11 defect types, from which 8 are distinct defect types. The number of defect types for each product varies between 1 and 3 defect types. The MAD dataset consists of multi-pose views of twenty LEGO toys (product classes), with up to three anomaly types. It has simulated and real images. The Real-IAD dataset consists of thirty product categories, up to four defect types per category, and a larger proportion of defect area and range of defect ratios than other datasets. We utilize single-view image data. The details are illustrated in Table 6.

We apply the default normalization of CLIP [28] to all datasets. After normalization, we resize the images to a resolution of $(518, 518)$ to obtain an appropriate visual feature map resolution.

Table 7. Defect-Aware text prompts for all defect types of the VisA dataset. *[cls]* represents a variable that takes as value all product classes in the VisA dataset.

Defect Type	Defect-Aware Text Prompts	Defect Type	Defect-Aware Text Prompts
Bent	"*[cls]* has a bent defect" "flawed *[cls]* with a bent lead" "a bend found in *[cls]*" "*[cls]* has a slight curve defect" "*[cls]* with noticeable bending" "a bent wire on *[cls]*"	Broken	"*[cls]* with a breakage defect" "broken *[cls]*" "*[cls]* with broken defect" "*[cls]* shows breakage" "broken or cracked areas on *[cls]*" "visible breakage on *[cls]*"
Bubble	"*[cls]* with bubbles defect" "bubbles seen on *[cls]*" "*[cls]* with bubble marks" "air bubbles in *[cls]*" "*[cls]* contains bubble defects" "small bubbles on *[cls]* surface"	Burnt	"*[cls]* with a burnt defect" "*[cls]* shows burn marks" "burnt areas on *[cls]*" "*[cls]* with signs of burning" "scorch marks on *[cls]*" "*[cls]* appears slightly burnt"
Chip	"*[cls]* with chip defect" "*[cls]* with fragment broken defect" "chipped areas on *[cls]*" "*[cls]* with chipped parts" "broken fragments on *[cls]*" "chip marks found on *[cls]*"	Crack	"*[cls]* with a crack defect" "*[cls]* has a visible crack" "cracked areas on *[cls]*" "*[cls]* with surface cracking" "fine cracks found on *[cls]*" "*[cls]* shows crack lines"
Damage	"*[cls]* has a damaged defect" "flawed *[cls]* with damage" "*[cls]* shows signs of damage" "damage found on *[cls]*" "*[cls]* with visible wear and tear" "*[cls]* with structural damage"	Extra	"*[cls]* with extra thing" "*[cls]* has a defect with extra thing" "extra material on *[cls]*" "*[cls]* contains additional pieces" "*[cls]* with extra component defect" "unwanted additions on *[cls]*"
Hole	"*[cls]* has a hole defect" "a hole on *[cls]*" "visible hole on *[cls]*" "*[cls]* has small punctures" "*[cls]* shows perforations" "hole present on *[cls]*"	Melded	"*[cls]* with melded defect" "melded parts on *[cls]*" "*[cls]* has fused areas" "fused spots on *[cls]*" "melded areas on *[cls]*" "*[cls]* with melded material"
Melt	"*[cls]* with melt defect" "melted areas on *[cls]*" "*[cls]* shows melting" "signs of melting on *[cls]*" "*[cls]* with melted spots" "*[cls]* has a melted appearance"	Missing	"*[cls]* with a missing defect" "flawed *[cls]* with something missing" "*[cls]* has missing parts" "missing components on *[cls]*" "absent pieces in *[cls]*" "*[cls]* is incomplete"
Partical	"*[cls]* with particles defect" "*[cls]* has foreign particles" "small particles on *[cls]*" "*[cls]* with unwanted particles" "contaminants found on *[cls]*" "*[cls]* with visible particles"	Scratch	"*[cls]* has a scratch defect" "flawed *[cls]* with a scratch" "scratches visible on *[cls]*" "*[cls]* has surface scratches" "small scratches found on *[cls]*" "*[cls]* with scratch marks"
Spot	"*[cls]* with spot defect" "spots visible on *[cls]*" "flawed *[cls]* with spots" "*[cls]* with visible spotting" "*[cls]* shows small spots" "surface spots on *[cls]*"	Stuck	"*[cls]* with a stuck defect" "*[cls]* stuck together" "*[cls]* has stuck parts" "adhesive issue causing *[cls]* to stick" "*[cls]* is partially stuck" "*[cls]* with adhesion defect"
Weird Wick	"*[cls]* with a weird wick defect" "*[cls]* has an unusual wick" "the wick on *[cls]* appears odd" "*[cls]* with a strangely shaped wick" "irregular wick found on *[cls]*" "odd wick defect on *[cls]*"	Wrong Place	"*[cls]* with defect that something on wrong place" "*[cls]* has a misplaced defect" "flawed *[cls]* with misplacing" "misaligned part on *[cls]*" "*[cls]* shows parts out of place" "misplacement detected on *[cls]*"

Table 8. Defect-Aware text prompts for all defect types of the MVTec-AD dataset. *[cls]* represents a variable that takes as value all product classes in the MVTec-AD dataset.

Defect Type	Defect-Aware Text Prompts	Defect Type	Defect-Aware Text Prompts
Bent	"*[cls]* has a bent defect" "flawed *[cls]* with a bent lead" "a bend found in *[cls]*" "*[cls]* has a slight curve defect" "*[cls]* with noticeable bending" "a bent wire on *[cls]*"	Broken	"*[cls]* has a broken defect" "flawed *[cls]* with breakage" "visible breakage on *[cls]*" "*[cls]* with broken areas" "*[cls]* shows signs of breaking" "cracked or broken spots on *[cls]*"
Color	"*[cls]* has a color defect" "inconsistent color on *[cls]*" "*[cls]* with color discrepancies" "*[cls]* has a noticeable color difference" "*[cls]* with irregular coloring" "*[cls]* has off-color patches"	Contamination	"*[cls]* has a contamination defect" "foreign particles on *[cls]*" "*[cls]* is contaminated" "*[cls]* contains contaminants" "*[cls]* has impurity issues" "traces of contamination on *[cls]*"
Crack	"*[cls]* has a crack defect" "a crack is present on *[cls]*" "cracked area on *[cls]*" "*[cls]* with noticeable cracking" "fine cracks found on *[cls]*" "*[cls]* shows surface cracks"	Cut	"*[cls]* has a cut defect" "cut marks on *[cls]*" "*[cls]* with visible cuts" "a cut detected on *[cls]*" "*[cls]* is sliced or cut" "surface cut seen on *[cls]*"
Damaged	"*[cls]* has a damaged defect" "flawed *[cls]* with damage" "*[cls]* with visible damage" "damaged areas on *[cls]*" "physical damage seen on *[cls]*" "noticeable wear on *[cls]*"	Fabric	"*[cls]* has a fabric defect" "*[cls]* has a fabric border defect" "*[cls]* has a fabric interior defect" "fabric quality issues on *[cls]*" "*[cls]* with textile irregularities" "fabric borders on *[cls]* show defects"
Faulty Imprint	"*[cls]* has a faulty imprint defect" "*[cls]* has a print defect" "incorrect printing on *[cls]*" "misaligned print on *[cls]*" "printing errors present on *[cls]*" "*[cls]* has a blurred print defect"	Glue	"*[cls]* has a glue defect" "*[cls]* has a glue strip defect" "excess glue on *[cls]*" "*[cls]* with uneven glue application" "*[cls]* has visible glue spots" "misplaced glue seen on *[cls]*"
Hole	"*[cls]* has a hole defect" "a hole on *[cls]*" "visible hole on *[cls]*" "*[cls]* with punctures" "small hole found in *[cls]*" "perforations present on *[cls]*"	Liquid	"*[cls]* has a liquid defect" "flawed *[cls]* with liquid" "*[cls]* with oil" "liquid marks on *[cls]*" "*[cls]* with liquid residue" "stains from liquid on *[cls]*"
Misplaced	"*[cls]* has a misplaced defect" "flawed *[cls]* with misplacing" "*[cls]* shows misalignment" "misplaced parts on *[cls]*" "*[cls]* with incorrect positioning" "positioning defects on *[cls]*"	Missing	"*[cls]* has a missing defect" "flawed *[cls]* with something missing" "*[cls]* has missing components" "missing parts on *[cls]*" "*[cls]* shows absent pieces" "certain parts missing from *[cls]*"
Poke	"*[cls]* has a poke defect" "*[cls]* has a poke insulation defect" "visible poke mark on *[cls]*" "*[cls]* has puncture marks" "a poke flaw on *[cls]*" "small poke defect on *[cls]*"	Rough	"*[cls]* has a rough defect" "rough texture on *[cls]*" "uneven surface on *[cls]*" "*[cls]* is coarser than expected" "surface roughness seen on *[cls]*" "texture defects on *[cls]*"
Scratch	"*[cls]* has a scratch defect" "flawed *[cls]* with a scratch" "visible scratches on *[cls]*" "*[cls]* with surface scratches" "minor scratches seen on *[cls]*" "*[cls]* shows scratch marks"	Squeeze	"*[cls]* has a squeeze defect" "flawed *[cls]* with a squeeze" "squeezed area on *[cls]*" "*[cls]* has compression marks" "*[cls]* appears squeezed" "flattened areas on *[cls]*"
Thread	"*[cls]* has a thread defect" "flawed *[cls]* with a thread" "loose threads on *[cls]*" "*[cls]* has visible threads" "untrimmed threads on *[cls]*" "threads sticking out on *[cls]*"		

Table 9. Defect-Aware text prompts for all defect types of the MPDD dataset. *[cls]* represents a variable that takes as value all product classes in the MPDD dataset.

Defect Type	Defect-Aware Text Prompts	Defect Type	Defect-Aware Text Prompts
Bent	"*[cls]* has a bent defect" "flawed *[cls]* with a bent lead" "a bend found in *[cls]*" "*[cls]* has a slight curve defect" "*[cls]* with noticeable bending" "a bent wire on *[cls]*"	Defective Painting	"*[cls]* with a defective painting defect" "flawed *[cls]* with painting imperfections" "*[cls]* has painting inconsistencies" "uneven painting on *[cls]*" "*[cls]* shows poor paint quality" "paint defects present on *[cls]*"
Flattening	"*[cls]* becomes flattened" "*[cls]* has a flatten defect" "flattening observed on *[cls]*" "*[cls]* appears compressed" "*[cls]* is flattened or squashed" "deformation detected on *[cls]*"	Hole	"*[cls]* with a hole defect" 'a hole on *[cls]*" 'visible hole in *[cls]*" "*[cls]* with puncture marks" 'hole detected in *[cls]*" "*[cls]* has small perforations"
Mismatch	"*[cls]* with bend and parts mismatch defec" "*[cls]* with parts mismatch defect" "*[cls]* has mismatched parts" "mismatched components on *[cls]*" "bend and parts misalignment in *[cls]*" "*[cls]* shows part misplacement"	Rust	"*[cls]* with a rust defect" "*[cls]* has rust patches" "rust spots on *[cls]*" "visible rust on *[cls]*" "*[cls]* shows signs of rusting" "*[cls]* affected by corrosion"
Scratch	"*[cls]* has a scratch defect" "flawed *[cls]* with a scratch' 'scratches visible on *[cls]*" "*[cls]* with surface scratches" "*[cls]* has scratch marks" "minor scratches found on *[cls]*"		

Table 10. Defect-Aware text prompts for all defect types of the MAD dataset. *[cls]* represents a variable that takes as value all product classes in the MAD dataset.

Defect Type	Defect-Aware Text Prompts	Defect Type	Defect-Aware Text Prompts
Burr	"*[cls]* has a burr defect" "sharp burr found on *[cls]*" "*[cls]* has excess material on edges" "burr formation detected on *[cls]*" "*[cls]* exhibits rough edges" " *[cls]* shows protruding material"	Missing	"*[cls]* has a missing defect" "flawed *[cls]* with something missing" "*[cls]* has missing components" "missing parts on *[cls]*" "*[cls]* shows absent pieces" "certain parts missing from *[cls]*"
Stain	"*[cls]* with a stain defect" "inconsistent color on *[cls]*" "*[cls]* with color discrepancies"		

Table 11. Defect-Aware text prompts for all defect types of the Real-IAD dataset. *[cls]* represents a variable that takes as value all product classes in the Real-IAD dataset.

Defect Type	Defect-Aware Text Prompts	Defect Type	Defect-Aware Text Prompts
Pit	"*[cls]* has a pit defect" "Small cavities or pits detected on *[cls]*" "*[cls]* with color discrepancies"	Scratch	"*[cls]* has a scratch defect" "flawed *[cls]* with a scratch' 'scratches visible on *[cls]*" "*[cls]* with surface scratches" "*[cls]* has scratch marks" "minor scratches found on *[cls]*"
Deformation	"*[cls]* has a deformation defect" "*[cls]* appears twisted or misshaped" "Structural distortion detected on *[cls]*" "Unexpected shape deformation found in *[cls]*" "*[cls]* exhibits rough edges" " *[cls]* shows signs of bending under stress"	Deformation	"*[cls]* has an abrasion defect" "*[cls]* has noticeable or scuffing" "*[cls]* is affected by continuous rubbing" "Worn or scraped areas found on *[cls]*"
Damaged	"*[cls]* has a damaged defect" "flawed *[cls]* with damage" "*[cls]* with visible damage" "damaged areas on *[cls]*" "physical damage seen on *[cls]*" "noticeable wear on *[cls]*"	Missing	"*[cls]* has a missing defect" "flawed *[cls]* with something missing" "*[cls]* has missing components" "missing parts on *[cls]*" "*[cls]* shows absent pieces" "certain parts missing from *[cls]*"
Foreign	"*[cls]* has foreign objects defect" "*[cls]* has a foreign defect" "Unexpected foreign material on *[cls]*" "*[cls]* contains an unwanted foreign object" "*[cls]* with extra thing" "*[cls]* has a defect with extra thing"	Contamination	"*[cls]* has a contamination defect" "foreign particles on *[cls]*" "*[cls]* is contaminated" "*[cls]* contains contaminants" "*[cls]* has impurity issues" "traces of contamination on *[cls]*"

Table 12. Detailed statistics on the MVTec-AD dataset.

Category	Product	Defects	Defect Type	Original Test	
				Anomalous	Normal
Objects	Bottle	Broken Large	Broken	20	20
		Broken Small	Broken	22	
		Contamination	Contamination	21	
	Cable	Bent Wire	Bent	13	58
		Cable Swap	Misplaced	12	
		Combined	Combined	11	
		Cut Inner Insulation	Cut	14	
		Cut Outer Insulation	Cut	10	
		Missing Cable	Missing	12	
		Missing Wire	Missing	10	
		Poke Insulation	Poke	10	
	Capsule	Crack	Crack	23	23
		Faulty Imprint	Faulty Imprint	22	
		Poke	Poke	21	
		Scratch	Scratch	23	
		Squeeze	Squeeze	20	
	Hazelnut	Crack	Crack	18	40
		Cut	Cut	17	
		Hole	Hole	18	
		Print	Faulty Imprint	17	
	Metal Nut	Bent	Bent	25	22
		Color	Color	22	
		Flip	Misplaced	23	
		Scratch	Scratch	23	
	Pill	Color	Color	25	26
		Combined	Combined	17	
		Contamination	Contamination	21	
		Crack	Crack	26	
		Faulty Imprint	Faulty Imprint	19	
		Pill Type	Damaged	9	
		Scratch	Scratch	24	
	Screw	Manipulated Front	Bent	24	41
		Scratch Head	Scratch	24	
		Scratch Neck	Scratch	25	
		Thread Side	Thread	23	
		Thread Top	Thread	23	
	Toothbrush	Defective	Damaged	12	30
	Transistor	Bent Lead	Bent	10	60
		Cut Lead	Cut	10	
		Damaged Case	Damaged	10	
		Misplaced	Misplaced	10	
	Zipper	Broken Teeth	Broken	19	32
		Combined	Combined	16	
		Fabric Border	Fabric	17	
		Fabric Interior	Fabric	16	
		Rough	Rough	17	
		Split Teeth	Misplaced	18	
		Squeezed Teeth	Squeezed	16	
Textures	Carpet	Color	Color	19	28
		Cut	Cut	17	
		Hole	Hole	17	
		Metal Contamination	Contamination	17	
		Thread	Thread	19	
	Grid	Bent	Bent	12	21
		Broken	Broken	12	
		Glue	Glue	11	
		Metal Contamination	Contamination	11	
		Thread	Thread	11	
	Leather	Color	Color	19	32
		Cut	Cut	19	
		Fold	Misplaced	17	
		Glue	Glue	19	
		Poke	Poke	18	
	Tile	Crack	Crack	17	33
		Glue Strip	Glue	18	
		Gray Stroke	Damaged	16	
		Oil	Liquid	18	
		Rough	Rough	15	
	Wood	Color	Color	8	19
		Combined	Combined	11	
		Hole	Hole	10	
		Liquid	Liquid	10	
		Scratch	Scratch	21	

Table 13. Detailed statistics on the VisA dataset. We relabeled every image originally marked as "combined" in the VisA dataset by identifying each individual defect it contains and assigning the image to all corresponding defect categories.

Category	Product	Defects	Defect Type	Test	
				Anomalous	Normal
Complex Structure	Pcb1	Bent	Bent	15	100
		Melt	Melt	52	
		Missing	Missing	20	
		Scratch	Scratch	21	
	Pcb2	Bent	Bent	15	100
		Melt	Melt	54	
		Missing	Missing	19	
		Scratch	Scratch	19	
	Pcb3	Bent	Bent	20	101
		Melt	Melt	41	
		Missing	Missing	20	
		Scratch	Scratch	25	
	Pcb4	Burnt	Burnt	8	101
		Scratch	Scratch	17	
		Dirt	Dirt	39	
		Damage	Damage	19	
		Extra	Extra	26	
		Missing	Missing	33	
		Wrong Place	Wrong Place	12	
Multiple Instances	Candle	Chunk of Wax Missing	Missing	15	100
		Damaged Corner of Packaging	Damaged	25	
		Different Colour Spot	Spot	22	
		Extra Wax in Candle	Extra	9	
		Foreign Particals on Candle	Particals	17	
		Wax Melded Out of the Candle	Melded	13	
		Weird Candle Wick	Weird Wick	11	
	Capsules	Bubble	Bubble	49	60
		Discolor	Discolor	15	
		Scratch	Scratch	15	
		Leak	Leak	20	
		Misheap	Damaged	20	
	Macaroni1	Chip Around Edge and Corner	Chip	25	100
		Different Colour Spot Similar Colour Spot	Spot	37	
		Small Cracks	Crack	14	
		Middle Breakage	Broken	10	
		Small Scratches	Scratches	27	
	Macaroni2	Breakage down the Middle	Broken	10	100
		Color Spot Similar to the Object Different Color Spot	Spot	35	
		Small Chip Around Edge	Chip	25	
		Small Cracks	Cracks	12	
		Small Scratches	Scratches	25	
Single Instance	Cashew	Burnt	Burnt	15	50
		Corner or Edge Breakage Middle Breakage	Broken	25	
		Different Colour Spot Same Colour Spot	Spot	25	
		Small Holes	Hole	21	
		Small Scratches	Scratch	16	
		Stuck Together	Stuck	6	
	Chewinggum	Chunk of Gum Missing Corner Missing	Missing	70	50
		Scratches	Scratch	14	
		Similar Colour Spot	Spot	25	
		Small Cracks	Crack	28	
	Fryum	Burnt	Burnt	9	50
		Corner or Edge Breakage Middle Breakage	Broken	30	
		Different Colour Spot Similar Colour Spot	Spot	36	
		Fryum Stuck Together	Stuck	20	
		Small Scratches	Scratch	9	
	Pipe Fryum	Burnt	Burnt	16	50
		Corner and Edge Breakage	Broken	25	
		Different Colour Spot Similar Colour Spot	Spot	31	
		Small Scratches	Scratch	22	
		Stuck Together	Stuck	10	
		Small Cracks	Crack	10	

Table 14. Detailed statistics on the MPDD dataset.

Product	Defects	Defect Type	Original Test	
			Anomalous	Normal
Bracket Black	Hole	Hole	12	32
	Scratches	Scratch	35	
Bracket Brown	Bend Mismatch	Mismatch	17	26
	Parts Mismatch	Mismatch	45	
Bracket White	Defective Painting	Defective Painting	13	30
	Scratches	Scratch	17	
Connector	Parts Mismatch	Mismatch	14	30
Metal Plate	Major Rust	Rust	14	26
	Scratches	Scratch	34	
	Total Rust	Rust	23	
Tubes	Anomalous	Flattening	69	32

Table 15. Detailed statistics on the MAD-real dataset.

Product	Defects	Defect Type	Original Test	
			Anomalous	Normal
Bear	Stains	Stains	24	5
Bird	Missing	Missing	22	5
Elephant	Missing	Missing	18	5
Parrot	Missing	Missing	23	5
Puppy	Stains	Stains	20	5
Scorpion	Missing	Missing	23	5
Turtle	Stains	Stains	21	5
Unicorn	Missing	Missing	21	5
Whale	Stains	Stains	32	5

Table 16. Detailed statistics on the MAD-sim dataset.

Product	Defects	Defect Type	Original Test	
			Anomalous	Normal
Bear	Burrs	Burrs	88	36
	Missing	Missing	112	
	Stains	Stains	59	
Bird	Burrs	Burrs	51	30
	Missing	Missing	160	
	Stains	Stains	40	
Cat	Burrs	Burrs	98	36
	Missing	Missing	151	
	Stains	Stains	58	
Elephant	Burrs	Burrs	72	36
	Missing	Missing	149	
	Stains	Stains	55	
Gorilla	Burrs	Burrs	67	20
	Missing	Missing	137	
	Stains	Stains	35	
Mallard	Burrs	Burrs	27	20
	Missing	Missing	157	
	Stains	Stains	33	
Obesobeso	Burrs	Burrs	101	36
	Missing	Missing	123	
	Stains	Stains	61	
Owl	Burrs	Burrs	41	30
	Missing	Missing	115	
	Stains	Stains	44	
Parrot	Burrs	Burrs	29	36
	Missing	Missing	131	
	Stains	Stains	42	
Pheonix	Burrs	Burrs	86	36
	Missing	Missing	150	
	Stains	Stains	69	
Pig	Burrs	Burrs	76	36
	Missing	Missing	138	
	Stains	Stains	70	
Puppy	Burrs	Burrs	63	36
	Missing	Missing	125	
	Stains	Stains	47	
Sabertooth	Burrs	Burrs	58	36
	Missing	Missing	136	
	Stains	Stains	47	
Scorpion	Burrs	Burrs	61	36
	Missing	Missing	121	
	Stains	Stains	53	
Sheep	Burrs	Burrs	39	36
	Missing	Missing	150	
	Stains	Stains	63	
Swan	Burrs	Burrs	66	36
	Missing	Missing	143	
	Stains	Stains	41	
Turtle	Burrs	Burrs	32	20
	Missing	Missing	130	
	Stains	Stains	35	
Unicorn	Burrs	Burrs	55	20
	Missing	Missing	132	
	Stains	Stains	35	
Whale	Burrs	Burrs	71	30
	Missing	Missing	127	
	Stains	Stains	53	
Zalika	Burrs	Burrs	56	36
	Missing	Missing	130	
	Stains	Stains	57	

Table 17. Detailed statistics on the Real-IAD dataset (Part I).

Product	Defects	Defect Type	Original Test	
			Normal	Anomalous
Audiojack	Deformation	Deformation	398	126
	Scratch	Scratch		4
	Missing	Missing		56
	Contamination	Contamination		27
Bottle Cap	Pit	Pit	369	65
	Scratch	Scratch		125
	Missing Parts	Missing Parts		1
	Contamination	Contamination		73
Button Battery	Pit	Pit	291	123
	Abrasion	Abrasion		68
	Scratch	Scratch		109
	Contamination	Contamination		117
End Cap	Scratch	Scratch	289	92
	Damage	Damage		119
	Missing Parts	Missing Parts		133
	Contamination	Contamination		80
Eraser	Pit	Pit	389	36
	Scratch	Scratch		101
	Missing Parts	Missing Parts		30
	Contamination	Contamination		68
Fire Hood	Pit	Pit	418	33
	Scratch	Scratch		51
	Missing Parts	Missing Parts		62
	Contamination	Contamination		23
Mint	Missing Parts	Missing Parts	305	111
	Foreign Objects	Foreign Objects		197
	Contamination	Contamination		142
Mounts	Pit	Pit	385	30
	Missing Parts	Missing Parts		131
	Contamination	Contamination		79
Pcb	Scratch	Scratch	278	103
	Missing Parts	Missing Parts		104
	Foreign Objects	Foreign Objects		129
	Contamination	Contamination		109
Phone Battery	Pit	Pit	349	38
	Scratch	Scratch		28
	Damage	Damage		125
	Contamination	Contamination		110
Plastic Nut	Pit	Pit	442	14
	Scratch	Scratch		13
	Missing Parts	Missing Parts		56
	Contamination	Contamination		35
Plastic Plug	Pit	Pit	368	121
	Scratch	Scratch		58
	Missing Parts	Missing Parts		31
	Contamination	Contamination		52
Porcelain Doll	Abrasion	Abrasion	402	64
	Scratch	Scratch		43
	Contamination	Contamination		89
Regulator	Scratch	Scratch	477	3
	Missing Parts	Missing Parts		63
Rolled Strip Base	Pit	Pit	250	170
	Missing Parts	Missing Parts		167
	Contamination	Contamination		172
Sim Card Set	Abrasion	Abrasion	305	148
	Scratch	Scratch		80
	Contamination	Contamination		168
Switch	Scratch	Scratch	266	164
	Missing Parts	Missing Parts		152
	Contamination	Contamination		161
Tape	Damage	Damage	397	128
	Missing Parts	Missing Parts		76
	Contamination	Contamination		21

Table 18. Detailed statistics on the Real-IAD dataset (Part II).

Product	Defects	Defect Type	Original Test	
			Normal	Anomalous
Terminalblock	Pit	Pit	308	142
	Missing Parts	Missing Parts		145
	Contamination	Contamination		106
Toothbrush	Abrasion	Abrasion	272	170
	Missing Parts	Missing Parts		137
	Contamination	Contamination		149
Toy	Pit	Pit	250	125
	Scratch	Scratch		127
	Missing Parts	Missing Parts		126
	Contamination	Contamination		126
Toy-brick	Pit	Pit	370	67
	Scratch	Scratch		60
	Missing Parts	Missing Parts		81
	Contamination	Contamination		53
Transistor1	Deformation	Deformation	265	171
	Missing Parts	Missing Parts		164
	Contamination	Contamination		134
U Block	Abrasion	Abrasion	436	20
	Scratch	Scratch		17
	Missing Parts	Missing Parts		44
	Contamination	Contamination		45
Usb	Deformation	Deformation	353	127
	Scratch	Scratch		54
	Missing Parts	Missing Parts		83
	Contamination	Contamination		39
Usb Adaptor	Pit	Pit	361	85
	Abrasion	Abrasion		22
	Scratch	Scratch		62
	Contamination	Contamination		111
Vcpill	Pit	Pit	398	50
	Scratch	Scratch		11
	Missing Parts	Missing Parts		107
	Contamination	Contamination		40
Wooden Beads	Pit	Pit	304	67
	Scratch	Scratch		96
	Missing Parts	Missing Parts		112
	Contamination	Contamination		117
Woodstick	Pit	Pit	442	7
	Scratch	Scratch		12
	Missing Parts	Missing Parts		69
	Contamination	Contamination		28
Zipper	Deformation	Deformation	250	125
	Damage	Damage		121
	Missing Parts	Missing Parts		125
	Contamination	Contamination		129

10. Baselines

To demonstrate the performance of MultiADS, we compare MultiADS with broad SOTA baselines. We run experiments for April-GAN [5], and other baseline results are taken from original papers. If the baseline does not report results for a specific dataset, then the results are taken from the latest publication, which includes these results. Details regarding each baseline are given as follows:

- PaDiM [7] utilizes a pre-trained Convolutional Neural Network (CNN) for patch embedding and multivariate Gaussian distributions to get a probabilistic representation for a one-class learning setting, the normal class. Also, it considers the semantic relations of CNN to improve the localization. Results are taken from [5, 38] baselines. Source code is available at https://github.com/taikiinoue45/PaDiM.
- CLIP [28] is a powerful zero-shot classification method. Results are taken from [44] baseline, and to perform the anomaly detection task, they use two classes of text prompt templates "*A photo of a normal [cls]*" and "*A photo of an anomalous [cls]*", where "*cls*" denotes the target class name. The anomaly score is computed according to Eq. [1] in the main manuscript. As for anomaly segmentation, they extend the above computation to local visual embedding to derive the segmentation. Source code is available at https://github.com/openai/CLIP.
- CLIP-AC [28] employs an ensemble of text prompt templates that are recommended for the ImageNet dataset [28]. Results are taken from [44] baseline, and they average the generated textual embeddings of normal and anomaly classes, respectively, and compute the probability and segmentation in the same way as CLIP. Source code is available at https://github.com/openai/CLIP.
- RegAD [13] is a few-shot learning approach that leverages feature registration as a category-agnostic approach. This approach trains a single generalizable model and does not require re-training or parameter fine-tuning for new categories. Results are taken from the original publication. Source code is available at https://github.com/MediaBrain-SJTU/RegAD.
- CoOp [42] is a representative method for prompt learning. Results are taken from [44] baseline for zero-shot setting and from [45] for few-shot setting. To adapt CoOp to zero- and few-shot anomaly detection, authors of [44, 45] replace its learnable text prompt templates $[V_1][V_2]\dots[V_N][cls]$ with normality and abnormality text prompt templates, where V_i is the learnable word embeddings. The normality text prompt template is defined as $[V_1][V_2]...[V_N][normal][cls]$, and the abnormality one is defined as $[V_1][V_2]\dots[V_N][anomalous][cls]$. Anomaly probabilities and segmentation are obtained in the same way as for AnomalyCLIP, and all parameters are kept the same as in the original paper. Source code is available at https://github.com/KaiyangZhou/CoOp.
- CoCoOp [41] extends the CoOp work by generalizing the learned context to wider unseen classes within the same dataset. CoCoOp learns a lightweight neural network to generate for each image an input-conditional token (vector), and the proposed dynamic prompts adapt to each instance and are less sensitive to class shift. Results are taken from [44] baseline. Source code is available at https://github.com/KaiyangZhou/CoOp.
- PatchCore [30] utilizes locally aggregated, mid-level patch features over a local neighborhood to ensure the retention of sufficient spatial context. PatchCore employs a memory bank for patch features to leverage nominal context at test time by using a greedy coreset subsampling. Results are taken from [5] baseline. Source code is available at https://github.com/amazon-science/patchcore-inspection
- WinCLIP [16] is a SOTA zero-shot anomaly detection method. Results for zero-shot settings are taken from the original publication and for few-shot settings are taken from [5] baseline. The authors design a large set of text prompt templates specific to anomaly detection and use a window scaling strategy to obtain anomaly segmentation. Source code is available at https://github.com/caoyunkang/WinClip.
- April-GAN [5] is an improved version of WinCLIP. We conducted experiments with this approach and all parameters are kept the same as in their paper. April-GAN first adjusts the text prompt templates and then introduces learnable linear projections to improve local visual semantics to derive more accurate segmentation. Source code is available at https://github.com/ByChelsea/VAND-APRIL-GAN.
- GraphCore [38] is a few-shot learning approach that utilizes memory banks to store image features. Results are taken from the original publication. They employ graph representation (Graph Neural Networks) to provide a visual isometric invariant feature (VIIF) as an anomaly measurement feature. The VIIF reduces the size of redundant features stored in memory banks. Results are taken from the original publication. The

authors have not provided a link to the source code yet.

- FastRecon [9] is a few-shot learning approach that utilizes a few normal samples as a reference to reconstruct its normal version, and sample alignment helps to detect anomalies. Thus, they propose a regression algorithm with distribution regularization for the transformation estimation. Results are taken from the original publication. Source code is available at https://github.com/FzJun26th/FastRecon.
- InCTRL [45] is a vision-language few-shot learning model that proposes an in-context residual learning approach. It aims to distinguish anomalies from normal samples by detecting residuals between test images and in-context few-shot normal sample prompts from the target domain on the fly. Results are taken from the original publication. Source code is available at https://github.com/mala-lab/InCTRL.
- PromptAD [25] is a vision-language few-shot learning approach that learns text prompts for anomaly detection. They propose to concatenate anomaly suffixes to transpose the semantics of normal prompts, in order to construct negative samples. They aim to control the distance between normal and abnormal prompt features through a hyperparameter. Results are taken from the original publication. Source code is available at https://github.com/FuNz-0/PromptAD.
- AnomalyCLIP [44] is a SOTA zero-shot anomaly detection method. Results are taken from the original publication. This approach learns a vector representation for text prompts for two states: normal and abnormal. They construct two templates of text prompts, object-aware text prompts and object-agnostic text prompts templates. Through an object-agnostic text prompt template, they aim to learn the shared patterns of different anomalies. Results are taken from the original publication. Source code is available at https://github.com/zqhang/AnomalyCLIP.

11. Experiments

In this section, we provide more details regarding our approach through ablation studies and the experiments that were conducted. We also visualize the results and discuss some insights and limitations of our approach.

11.1. Experiment Details

In this subsection, we detail the experimental setup. We use the ViT-L-14-336 CLIP backbone from OpenCLIP [15], pre-trained on the LAION-400M_E32 setting of open-clip. The learning rate is set to 0.001, with a batch size of 8. The stage number $m = 4$. The features are selected from layers 6, 12, 18, and 24.

We adopt a transfer learning setting, training the model on one dataset and evaluating it on the remaining. Specifically, we train our model on MVTec-AD and evaluate it on VisA, MPDD, MAD, and Real-IAD, as well as train on VisA and evaluate on MVTec-AD. Other combinations are not included in the results, as most baselines focus on the aforementioned configurations. During training, we exclude all images labeled with "combined" defects, which indicate multiple defects in a single image. This exclusion is due to the datasets providing binary anomaly masks that treat all defects as identical. Since combined defects are relatively rare in the datasets (see Tables 12, 13, 14), we opted to leave them out during training. However, for testing, all images with multiple defects are included to ensure a fair comparison.

11.2. Ablation Studies

Here, we will give more details regarding our ablation studies and show additional results of the experiments we have conducted for the multi-type anomaly segmentation (MTAS) task, binary zero-/few-shot anomaly detection task, and zero-batch task.

11.2.1. Global Anomaly Score

To assess the impact of the global anomaly score on anomaly detection, we conducted ablation studies using our MultiADS model without the global anomaly score, referred to as MultiADS-L. As shown in Table 19, removing the global anomaly score leads to a noticeable performance drop in the zero-shot setting. However, the performance drop in the few-shot setting is minimal, likely because the additional information provided by the test data compensates for the absence of global context.

11.2.2. Defect-Aware Text Prompts

To show the importance of the defect-aware text prompts, we conduct experiments on the MPDD dataset with our approach, MultiADS. First, we train our model on the MVTec-AD dataset, with defect-aware text prompts constructed for the MVTec-AD dataset. Then, during the testing phase, instead of using the defect-aware text prompts constructed for the MPDD dataset, we use defect-aware text prompts constructed for the

Table 19. Ablation study for testing without global anomaly score. MultiADS is our proposed method, while MultiADS-L is the ablated version without including the global anomaly score.

Settings	Training → Testing	Method	Image-Level		
			AUROC	F1-max	AP
Zero-shot	MVTec-AD → VisA	MultiADS	83.6	80.3	86.9
		MultiADS-L	82.1 (+1.5)	80.3 (+0.0)	85.8 (+1.1)
	MVTec-AD → MPDD	MultiADS	78.3	79.2	78.4
		MultiADS-L	76.5 (+1.8)	79 (+0.2)	78.1 (+0.3)
Few-shot (k=4)	MVTec-AD → VisA	MultiADS	93.3	89.7	94.3
		MultiADS-L	93.8 (-0.5)	89.6 (+0.1)	94.5 (-0.2)
	MVTec-AD → MPDD	MultiADS	86	87.2	89.4
		MultiADS-L	85.6 (+0.4)	86.8 (+0.4)	89.3 (+0.1)

Table 20. Ablation Study: Results for MultiADS for each product of the MPDD dataset with different defect-aware text prompts from the VisA dataset and the MPDD dataset on few-shot (k=1) anomaly detection and segmentation tasks. Our model is trained on the MVTec-AD dataset. (**Bold** represents the best performer)

Setting	k=1													
MVTec → MPDD	Pixel-Level								Image-Level					
Product	AUROC		F1-max		AP		AUPRO		AUROC		F1-max		AP	
	VisA	MPDD	VisA	MPDD	VisA	MPDD	VisA	MPDD	VisA	MPDD	VisA	MPDD	VisA	MPDD
Bracket_black	96.7	97.2	11.2	18.7	4.5	11.8	88	89.5	63.4	74.6	78.5	81.6	68.6	80.8
Bracket_brown	96	96.2	14.9	17.6	7.5	8.7	91	91.1	60.4	53.3	80	79.7	72.5	71.4
Bracket_white	99.7	99.7	20.7	24.5	12.8	15.2	96.5	96.7	73.4	81.1	75	78.3	77	82.5
Connector	95.9	96.4	35.3	33.9	33.7	32.4	87.2	87.8	92.9	91.4	78.8	82.8	88.9	9.3
Metal_plate	96.3	96.3	74.6	73.1	81.2	74.8	90.6	89.8	99	92	97.9	90.1	99.6	97.2
Tubes	98.7	98.8	69	68.7	71	70.4	95	95.5	97.3	97.6	96.4	95.5	99	99.1
Average	97.2	**97.4**	37.6	**39.4**	35.1	**35.6**	91.4	**91.7**	81.1	**81.7**	84.4	**84.6**	84.3	**86.7**

VisA dataset. The results are shown in Table 20. We observe that our approach, MultiADS, performs quite well even when we utilize the defect-aware text prompts of the other dataset for all the metrics on pixel-level and image-level on few-shot anomaly detection and segmentation tasks. Also, we note that to achieve the best performance, especially on the image level, it is crucial to employ defect-aware text prompts suitable for the products of the testing dataset, the MPDD dataset.

In addition to the results shown in the main manuscript, in Table 2 we list the segmentation performance for some sample defect types that are seen/unseen during the training phase. We notice that defects such as *stains* and *scratches* are easy to locate and classify, as they also occur on the training dataset - MVTec-AD. For unseen defects like *burrs* and *mismatch*, our model achieves slightly lower accuracy. On the other hand, for other unseen defects such as *flattening*, we perform with high precision for the classification task. These results, similar to results in the main manuscript, reflect that our approach, MultiADS, has generalization ability on large and complex datasets and unseen defects in the training dataset.

Table 21. Results MTAS for zero-shot setting at pixel-level for sample defect-types. The model is trained on the MVTec-AD dataset. - indicates **unseen** defect types while ✓indicates **seen** defect types during training.

(a) MAD-sim

	Defects	AUROC	F1-Score	AP
-	Burrs	95.56	1.18	1.67
✓	Missing	86.52	2.56	3.08
✓	Stains	98.19	15.02	9.92

(b) MPDD

	Defects	AUROC	F1-Score	AP
-	Mismatch	88.44	2.56	1.04
-	Flattening	96.72	36.06	8.33
✓	Scratch	96.67	26.99	20.26

11.2.3. Batched Zero-shot Setting

The idea behind the batched zero-shot setting is to utilize all text samples in X_{test} without relying on any labels. This approach can be viewed as a form of domain adaptation, enabling the trained model to better align with the target domain. Inspired by the methodology proposed

Table 22. Image level results for batched zero-shot setting. All results are AUROC values (%). The numbers of baselines are taken from AnomalyDINO [6]. 448 and 672 are the resolutions of the input image.

Setting	Method	MVTec	VisA
Batched zero-shot	ACR [21]	85.8	/
	MuSc [24]	**97.8**	92.8
	AnomalyDINO$_{(448)}$ [6]	93.0	89.7
	AnomalyDINO$_{(672)}$ [6]	94.2	90.7
	MultiADS (ours)	96.1	**93.1**

by AnomalyDINO [6], we employ a memory bank to facilitate this adaptation process. For each test sample $x^{(k)} \in X_{\text{test}}$, let $\mathbf{Z}_i^k \in \mathbb{R}^{h \times w \times N_z}$ denote the adapted image patch embeddings at state i for given image $x^{(k)}$. We define memory bank $\mathcal{M}_i$ as the union of all image patch embeddings at stage i across the entire text set X_{test}:

$$\mathcal{M}_i = \bigcup_{x^{(k)} \in X_{\text{test}}} \left\{ \mathbf{Z}_i^k[a,b] | a \in [h], b \in [w] \right\}. \quad (7)$$

During testing, for each given image $x^{(k)}$, we compute the cosine similarity between its adapted image patch embedding $\mathbf{Z}_i^k[a,b] \in \mathbb{R}^{N_z}$ and all embeddings in the memory bank $\mathcal{M}_i \setminus \mathbf{Z}_i^k[a,b]$. Since the memory bank may include anomalous features (due to the unlabeled setting), directly selecting the nearest neighbor might not reliably represent nominal behavior. To address this, and based on the assumption that most patches in the memory bank are nominal, we replace the nearest neighbor with the k-th nearest neighbor, where k corresponds to the α-quantile of the similarity scores. Thus, the set of cosine similarity scores is defined as follows:

$$\mathcal{D}\left(\mathbf{Z}_i^k[a,b],\, \mathcal{M}_i \setminus \{\mathbf{Z}_i^k[a,b]\}\right) = \left\{ d\left(\mathbf{Z}_i^k[a,b], \mathbf{x}\right) \mid \mathbf{x} \in \mathcal{M}_i \setminus \{\mathbf{Z}_i^k[a,b]\} \right\}. \quad (8)$$

where $d(\cdot)$ represents the cosine similarity. The reference anomaly score for image patch embedding $\mathbf{Z}_i^k[a,b]$ is defined as follows:

$$s(\mathbf{Z}_i^k[a,b]) = q_\alpha(\mathcal{D}(\mathbf{Z}_i^k[a,b], \mathcal{M}_i \setminus \mathbf{Z}_i^k[a,b])), \quad (9)$$

where q_α is the α quantile of the similarity score set. The comparison of our MultiADS approach with other baselines is listed in Table 22.

11.2.4. Backbones

In Table 23, we show the impact of different architectures and resolutions for our proposed approach, MultiADS. To evaluate the performance of our proposed approach, MultiADS, and other baselines, we perform zero-shot and few-shot anomaly detection and segmentation on five datasets, MVTec-AD [1], VisA [46], MPDD [17], MAD [43], and Real-IAD [37]. Results of other baselines are taken from the original published papers or the most recent publications. Thus, for some of the baselines, we are missing the evaluation with different metrics, such as F1-max, AP, and AUPRO on pixel-level, or F1-max and AP for image-level.

11.2.5. Additional Results

In Tables 24, 25, and 26, we show results for our approach, MultiADS, and other baselines on a few-shot setting with $k \in [1, 2, 4, 8]$ on anomaly detection and segmentation tasks on three datasets, VisA, MPDD, and MVTec-AD, respectively. In Tables 27, 28, and 29, we show results for our approach, MultiADS, on a few-shot setting with $k \in \{1, 2\}$ on anomaly detection and segmentation tasks for each product of the VisA, MPDD, and MVTec-AD datasets, respectively. In Tables 30 and 31, we show results for the variant of our approach, MultiADS-F, on the few-shot setting with $k \in \{1, 2\}$ on anomaly detection and segmentation tasks for each product of the VisA and MPDD datasets, respectively.

Furthermore, in Table 32, we show results for our proposal, MultiADS, and the most recent baseline, AdaCLIP, for all products of the Real-IAD dataset. We note that our proposal outperforms AdaCLIP for all metrics, and the largest improvement of our method is at the image level. Similarly, in Table 33, we show results for our proposal, MultiADS, and the most competitive baseline, April-GAN, for all products of the MAD dataset. We note that our proposal overall outperforms April-GAN for almost all metrics, and the largest improvement of our method is at the pixel level.

11.3. Visualizations

In this subsection, we present additional visualizations of our anomaly segmentation results. We include eight examples of products from the MVTec-AD, VisA, and MPDD datasets: hazelnut (Figure 7), screw (Figure 8), and leather (Figure 9) from MVTec-AD; pipe_fryum (Figure 10), and capsule (Figure 11) from VisA; and connector (Figure 12) and tube (Figure 13) from MPDD. All segmentation visualizations are performed in a few-shot ($k = 4$) setting. Specifically, the models for hazelnut, screw, and leather were trained on the VisA dataset; the models for pipe_fryum, capsule, and candle were trained on the MVTec-AD dataset; and the models for connector and tube were trained on the MVTec-AD dataset. We discuss some insights and limitations in the caption of these figures.

Table 23. Ablation study for training and testing with different architectures/resolutions for BADS. MultiADS applies the ViT-L-14 architecture with a resolution of 336.

Settings	Dataset	Architecture	Resolution	Image-Level		
				AUROC	F1-max	AP
Zero-shot	VisA	ViT-B-16	224	74	76.6	79
		ViT-B-32	224	68.4	74.6	73.5
		ViT-L-14	224	75.2	78.4	80.6
		ViT-L-14	336	83.6	80.3	86.9
	MPDD	ViT-B-16	224	67.7	77.2	74.4
		ViT-B-32	224	60.7	75	68.8
		ViT-L-14	224	71.6	77.8	76.8
		ViT-L-14	336	78.3	79.2	78.4
Few-shot (k=4)	VisA	ViT-B-16	224	90	86	91.9
		ViT-B-32	224	83.1	81.4	85.4
		ViT-L-14	224	92	88	93.5
		ViT-L-14	336	93.3	89.7	94.3
	MPDD	ViT-B-16	224	80.2	81.6	80
		ViT-B-32	224	78.2	83.1	80.2
		ViT-L-14	224	82	82.9	84.3
		ViT-L-14	336	85.6	87.2	89.4

Table 24. Few-shot anomaly detection and segmentation on the VisA Datasets. April-GAN baseline and our model are trained on the MVTec-AD dataset. (- denotes the results for this metric are not reported in the original paper; **bold** represents the best performer)

Settings		**k=1**					**k=2**				
VisA		Pixel-Level		Image-Level			Pixel-Level		Image-Level		
Method	Venue	AUROC	AUPRO	AUROC	F1-max	AP	AUROC	AUPRO	AUROC	F1-max	AP
PaDiM	ICPR21	89.9	64.3	62.8	75.3	68.3	92.0	70.1	67.4	75.7	71.6
CoOp	IJCV22	-	-	-	-	-	-	-	83.5	-	-
PatchCore	CVPR23	95.4	80.5	79.9	81.7	82.8	96.1	82.6	81.6	82.5	84.8
WinCLIP	CVPR23	96.4	85.1	83.8	83.1	85.1	96.8	86.2	84.6	83.0	85.8
April-GAN	CVPR23	96.0	90.0	91.2	86.9	93.3	96.2	90.1	92.2	87.7	94.2
PromptAD	CVPR24	96.7	-	86.9	-	-	97.1	-	88.3	-	-
InCTRL	CVPR24	-	-	-	-	-	-	-	87.7	-	-
AnomalyGPT	AAAI24	96.2	-	87.4	-	-	96.4	-	88.6	-	-
MultiADS (ours)		**97.1**	**92.7**	91.9	**88.3**	93.1	**97.2**	**93.1**	**93.3**	**89.5**	93.9
MultiADS-F (ours)		96.6	91.7	**92**	88.1	**93.9**	96.7	91.9	92.8	88.5	**94.4**

Settings		**k=4**					**k=8**				
VisA		Pixel-Level		Image-Level			Pixel-Level		Image-Level		
Method	Venue	AUROC	AUPRO	AUROC	F1-max	AP	AUROC	AUPRO	AUROC	F1-max	AP
PaDiM	ICPR21	93.2	72.6	72.8	78.0	75.6	-	-	78.1	-	-
CoOp	IJCV22	-	-	84.2*	-	-	-	-	84.8	-	-
PatchCore	CVPR23	96.8	84.9	85.3	84.3	87.5	-	-	87.3	-	-
WinCLIP	CVPR23	97.2	87.6	87.3	84.2	88.8	-	-	88.0	-	-
April-GAN	CVPR23	96.2	90.2	92.6	88.4	94.5	96.3	90.2	92.7	88.5	94.6
PromptAD	CVPR24	**97.4**	-	89.1	-	-	-	-	-	-	-
InCTRL	CVPR24	-	-	90.2*	-	-	-	-	90.4	-	-
AnomalyGPT	AAAI24	96.7	-	90.6	-	-	-	-	-	-	-
MultiADS (ours)		96.9	91.1	**93.3**	**89.7**	94.3	**97.4**	**93.5**	**94.7**	**91.3**	94.9
MultiADS-F (ours)		97.0	**91.5**	92.8	88.5	**94.6**	96.9	92.1	93.8	89.5	**95.1**

Table 25. Few-shot anomaly detection and segmentation on the MPDD Dataset. April-GAN baseline and our model are trained on the MVTec-AD dataset. (- denotes the results for this metric are not reported in the original paper; **bold** represents the best performer)

Settings		**k=1**					**k=2**				
MPDD		Pixel-Level		Image-Level			Pixel-Level		Image-Level		
Method	Venue	AUROC	AUPRO	AUROC	F1-max	AP	AUROC	AUPRO	AUROC	F1-max	AP
PaDiM	ICPR21	73.9	-	57.5	-	-	75.4	-	58.0	-	-
RegAD	ECCV22	92.6	-	60.9	-	-	93.2	-	63.4	-	-
PatchCore	CVPR22	79.4	-	68.9	77.2	-	84.4	-	75.5	81.7	-
April-GAN	CVPR23	96.9	91.4	84.6	**86.8**	**88.6**	96.9	91.4	84.6	**86.8**	88.6
GraphCore	ICLR23	95.2	-	**84.7**	-	-	95.4	-	85.4	-	-
FastRecon	ICCV23	96.4	-	72.2	79.1	-	96.7	-	76.1	82.8	-
MultiADS (ours)		97.4	91.7	81.7	84.6	86.7	97.7	**92.4**	**86.6**	86.6	**90.1**
MultiADS-F (ours)		**97.7**	**92.2**	80.1	82.5	84	**97.8**	**92.4**	83.8	85.8	86.9
Settings		**k=4**					**k=8**				
MPDD		Pixel-Level		Image-Level			Pixel-Level		Image-Level		
Method	Venue	AUROC	AUPRO	AUROC	F1-max	AP	AUROC	AUPRO	AUROC	F1-max	AP
PaDiM	ICPR21	75.9	-	58.3	-	-	76.2	-	58.5	-	-
RegAD	ECCV22	93.9	-	68.8	-	-	95.1	-	71.9	-	-
PatchCore	CVPR22	92.8	-	77.8	82.4	-	92.8	-	77.8	82.4	-
April-GAN	CVPR23	96.9	91.4	84.6	86.8	88.6	96.7	91	86	**87.8**	**90.8**
GraphCore	ICLR23	95.7	-	85.7	-	-	95.9	-	86.0	-	-
FastRecon	ICCV23	97.2	-	79.3	83.5	-	97.2	-	79.3	83.5	-
MultiADS (ours)		97.5	94.1	84.3	84.8	87.2	97.7	**93.1**	83.3	87.6	88.1
MultiADS-F (ours)		**97.8**	**94.4**	86.2	**88.5**	**88.8**	**98**	92.8	85	85.2	89.1

Table 26. Few-shot anomaly detection and segmentation on the MVTec-AD Dataset. April-GAN baseline and our model are trained on the VisA dataset. (- denotes the results for this metric are not reported in the original paper; **bold** represents the best performer)

Settings		**k=1**					**k=2**				
MVTec-AD		Pixel-Level		Image-Level			Pixel-Level		Image-Level		
Method	Venue	AUROC	AUPRO	AUROC	F1-max	AP	AUROC	AUPRO	AUROC	F1-max	AP
PaDiM	ICPR21	89.9	64.3	62.8	75.3	68.3	92.0	70.1	67.4	75.7	71.6
PatchCore	CVPR23	95.4	80.5	79.9	81.7	82.8	96.1	82.6	81.6	82.5	84.8
WinCLIP	CVPR23	**96.4**	85.1	83.8	83.1	85.1	**96.8**	86.2	84.6	83.0	85.8
April-GAN	CVPR23	96.0	90.0	91.2	86.9	93.3	96.2	90.1	92.2	87.7	94.2
PromptAD	CVPR24	96.7	-	86.9	-	-	97.1	-	88.3	-	-
AnomalyGPT	AAAI24	96.2	-	87.4	-	-	96.4	-	88.6	-	-
MultiADS (ours)		93.2	**90.6**	**93**	**94**	**96.4**	93.2	**90.8**	**93.5**	**94.5**	**96.6**
Settings		**k=4**					**k=8**				
MVTec-AD		Pixel-Level		Image-Level			Pixel-Level		Image-Level		
Method	Venue	AUROC	AUPRO	AUROC	F1-max	AP	AUROC	AUPRO	AUROC	F1-max	AP
PaDiM	ICPR21	93.2	72.6	72.8	78.0	75.6	-	-	-	-	-
PatchCore	CVPR23	96.8	84.9	85.3	84.3	87.5	-	-	-	-	-
WinCLIP	CVPR23	97.2	87.6	87.3	84.2	88.8	-	-	-	-	-
April-GAN	CVPR23	95.9	**91.8**	92.8	92.8	96.3	**96.1**	**92.2**	93.3	93.1	96.5
PromptAD	CVPR24	**97.4**	-	89.1	-	-	-	-	-	-	-
AnomalyGPT	AAAI24	96.7	-	90.6	-	-	-	-	-	-	-
MultiADS (ours)		93.3	90.9	**96.6**	**95.4**	**98.1**	93.4	91.2	**97.2**	**96**	**98.5**

Table 27. Results for MultiADS for each product of the VisA dataset on few-shot anomaly detection and segmentation tasks. Our model is trained on the MVTec-AD dataset.

Settings	k=1							k=2						
VisA	Pixel-Level				Image-Level			Pixel-Level				Image-Level		
Product	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP
Candle	98.7	39.7	25.2	97	91.2	88.1	90.8	98.7	39.3	24.7	97.1	92	88.8	91
Capsules	98.1	47.1	39.9	90.7	95.4	92.1	97.6	98.3	48.8	44.2	92.9	96.5	92.5	98.1
Cashew	94.6	49.3	41.8	96.3	91	89.7	95.5	94.3	49.5	41.4	96.5	95	92.2	97.6
Chewinggum	99.7	72.4	76.1	95.1	98.4	97	99.4	99.6	71.1	73.6	94.7	98.4	96.4	99.3
Fryum	95	35.4	29.8	93	96.6	92.9	98.3	95.1	36.7	30.7	93.3	97.3	95.9	98.9
Macaroni1	99.5	33.6	26.2	95.6	90.8	84	92.9	99.5	30.1	22.8	96.1	90.6	83.7	92.3
Macaroni2	98.7	26.8	14.1	90.4	85.8	80.2	89.2	98.8	23.8	12.5	89.6	83	75.6	85.6
Pcb1	96.6	36.1	29.9	93.2	94.9	90.6	94.1	97	42.5	36.2	93.5	93.5	88.6	92.3
Pcb2	95.4	27.4	19.1	84.7	77.4	72.7	78.5	95.6	35.9	24.9	86.3	87.5	82.7	87.4
Pcb3	93.8	42.9	32.4	86.5	86.4	81.3	87.4	94.1	50.1	39.8	87.3	90.9	84	91.2
Pcb4	96.6	38.3	34	91.9	96.4	93.8	94.5	96.7	39.6	34.3	92.1	96.1	93.7	93.3
Pipe_fryum	98.1	50.1	40.8	97.8	98.9	97.5	99.3	98.1	51.1	41	97.9	99	99.5	99.3
Average	97.1	41.6	34.1	92.7	91.9	88.3	93.1	97.2	43.2	35.5	93.1	93.3	89.5	93.9

Table 28. Results for MultiADS for each product of the MPDD dataset on few-shot anomaly detection and segmentation tasks. Our model is trained on the MVTec-AD dataset.

Settings	k=1							k=2						
MPDD	Pixel-Level				Image-Level			Pixel-Level				Image-Level		
Product	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP
Bracket_black	97.2	18.7	11.8	89.5	74.6	81.6	80.8	98.3	35	25.3	94.3	82.4	82.1	88.9
Bracket_brown	96.2	17.6	8.7	91.1	53.3	79.7	71.4	96.2	19.9	11.1	90.1	65.8	81	78.1
Bracket_white	99.7	24.5	15.2	96.7	81.1	78.3	82.5	99.6	23.7	14.1	96.2	84.1	81.1	85
Connector	96.4	33.9	32.4	87.8	91.4	82.8	89.3	96.2	35.1	34.3	87.7	93.8	85.7	91
Metal_plate	96.3	73.1	74.8	89.8	92	90.1	97.2	96.8	75	77.8	90.7	95.7	93.7	98.5
Tubes	98.8	68.7	70.4	95.5	97.6	95.5	99.1	98.8	69.2	71.2	95.7	97.9	96.3	99.2
Average	97.4	39.4	35.6	91.7	81.7	84.6	86.7	97.7	43	39	92.4	86.6	86.6	90.1

Table 29. Results for MultiADS for each product of the MVTec-AD dataset on few-shot anomaly detection and segmentation tasks. Our model is trained on the VisA dataset.

Settings	k=1							k=2						
MVTec-AD	Pixel-Level				Image-Level			Pixel-Level				Image-Level		
Product	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP
Bottle	93.3	63.2	66.9	89.3	97.2	96.7	99.2	93.4	63.6	67.3	89.3	96.9	96.7	99.1
Cable	84.8	37.3	34.1	81	82.7	80.8	90.3	83.8	39.8	35.1	80.6	84.6	82.2	91
Capsule	95.3	36.6	31.1	93.6	73.6	93.4	91.6	95.4	36.7	30.6	94	72.9	93	91.4
Carpet	99.1	73.1	78	97.3	99.7	98.3	99.9	99.1	72.9	77.6	97.6	99.8	98.9	99.9
Grid	98.3	45.3	40.7	94.5	95.8	96.5	98.1	98.6	45.6	42.6	95.1	97.7	97.4	98.9
Hazelnut	98	61	63.9	96	99.8	99.3	99.9	98.2	63.1	66.4	96.2	98.9	97.9	99.3
Leather	99.6	59.3	60.8	99.2	98.9	99.5	99.6	99.6	59.1	61	99.2	100	100	100
Metal_nut	83.8	40.9	43.6	85.5	97.1	96.8	99.3	83.8	41.5	45	85.8	99.7	98.4	99.9
Pill	88.8	40.4	38.6	96.3	96.4	96.9	99.2	88.6	40.3	38.2	96.3	95.5	97.2	99
Screw	98	34.7	28.6	93.3	78.8	87.5	91.2	98	35.5	31.1	93.3	76.9	86.5	91.3
Tile	95.2	69.6	64	91.7	98	96.4	99.2	95.2	69.6	64.1	91.4	98.4	97	99.3
Toothbrush	98.1	59.2	56	95.6	99.7	98.4	99.9	98	58.7	56.4	95.5	99.7	98.4	99.9
Transistor	71.4	25	22.9	59.1	82.8	75.4	80.1	72.4	27.1	24.5	59.8	85	78.6	81.2
Wood	96.4	67.9	68.8	95.7	99.1	97.4	99.7	96.5	68.1	69.3	95.8	99.3	97.5	99.8
Zipper	97.2	63.8	63.1	91.2	95.9	96.3	98.8	97.3	64.8	64	91.4	97.4	97.1	99.3
Average	93.2	51.8	50.7	90.6	93	94	96.4	93.2	52.4	51.5	90.8	93.5	94.5	96.6

Table 30. Results for MultiADS-F for each product of the VisA dataset on few-shot anomaly detection and segmentation tasks. Our model is trained on the MVTec-AD dataset.

Settings	k=1							k=2						
VisA	Pixel-Level				Image-Level			Pixel-Level				Image-Level		
Product	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP
Candle	98.7	40.4	27.1	97.1	90.4	84.4	91	98.7	40	26.7	97	90.6	85.7	91.1
Capsules	97.6	47.2	40.6	88.1	93.1	91.1	96.6	97.7	48.2	42.3	89.6	93.8	89.7	96.8
Cashew	94.1	39.4	32.1	96.6	91.7	89.2	95.7	93.9	39.9	31.6	96.6	94.3	91.3	97.3
Chewinggum	99.6	77.6	82.2	93.1	98.9	97.4	99.5	99.6	77.4	81.9	93.1	98.3	97.4	99.3
Fryum	94.3	33.3	27	92	93.8	93.3	97.4	94.4	34.1	27.5	92.3	94.7	93.8	98
Macaroni1	99.5	35.7	26	96.2	89.1	82.4	91.7	99.5	35	24.5	96.4	90.3	82.4	92.5
Macaroni2	98.8	26.8	14.3	89.8	84.3	77.9	88.7	98.8	25.5	13.7	89.3	82.8	77.2	86.3
Pcb1	95.2	23.2	17.3	92	95.8	89.3	96.2	95.7	25	19.1	92.3	94.9	87.1	95.4
Pcb2	94.4	31	21.6	82.3	83.7	78.8	85.7	94.5	35	24.4	83.3	87.9	80.4	90.2
Pcb3	93.5	39.9	29.9	83.6	86.1	80.4	88	93.7	46.1	35.5	84	89.6	83	90.5
Pcb4	96.5	39.7	35.1	91.6	97.5	94.1	96.7	96.5	40.5	35.4	91.6	97.4	94.2	96.5
Pipe_fryum	97.4	43.4	34.3	97.7	99.1	99	99.4	97.4	43	33.9	97.6	99	99.5	99.3
Average	96.6	39.8	32.3	91.7	92	88.1	93.9	96.7	40.8	33	91.9	92.8	88.5	94.4

Table 31. Results for MultiADS-F for each product of the MPDD dataset on few-shot anomaly detection and segmentation tasks. Our model is trained on the MVTec-AD dataset.

Settings	k=1							k=2						
MPDD	Pixel-Level				Image-Level			Pixel-Level				Image-Level		
Product	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP
Bracket_black	97.6	25	18.2	91.8	73.1	77.1	82.8	98.1	32.1	23.7	94.1	78.6	81.1	86.2
Bracket_brown	95.9	18.5	9.8	88.9	54.6	79.7	74.4	95.9	21.1	13.4	87.9	65.4	81	80.6
Bracket_white	99.6	22.2	14.1	95.8	74.6	78.9	69.8	99.6	22.4	12.8	95.4	75.4	81.1	70.4
Connector	96.3	30.8	27.3	87.3	84.8	70.6	79.8	96	31.8	28.6	86.9	89	82.8	86.7
Metal_plate	97.6	80.4	78.3	93.2	98.4	97.3	99.4	98.1	82.5	81.4	94.2	98.9	97.3	99.6
Tubes	99	65.6	68.9	96	95.4	91.5	98.1	99	66.2	69.5	96.2	95.3	91.4	98
Average	97.7	40.4	36.1	92.2	80.1	82.5	84	97.8	42.7	38.2	92.4	83.8	85.8	86.9

Table 32. Results for MultiADS and the most recent baseline approach, AdaCLIP, for each product of the Real-IAD dataset on few-shot (k=4) anomaly detection and segmentation tasks. Both models are trained on the MVTec-AD dataset.

Baseline	MultiADS							AdaCLIP						
Real-IAD	Pixel-Level				Image-Level			Pixel-Level				Image-Level		
Product	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP
Audiojack	98.4	54.6	49.9	89.3	75.8	72.8	77.8	97.21	42.47	37.46	-	66.2	53.68	57.39
Bottle Cap	99	41.5	34.9	92	81	71.5	81.3	98.4	34.8	30.06	-	86.84	76.87	80.65
Button Battery	97.5	47.7	46.7	89.3	72.9	75.4	82	96.69	45.7	45.98	-	69.47	74.45	78.94
End Cap	96	30.6	21.7	86.8	77.3	76.8	84.4	90.59	17.74	7.89	-	60.45	74.85	67.59
Eraser	99.8	62.2	63.8	98.6	92.2	86.2	92.5	99.09	59.5	59.52	-	71.49	60.43	67.37
Fire hood	99.5	57.2	58.6	97.8	94.1	81.5	87.5	99.36	51.82	54	-	87.76	72.36	73.05
Mint	97.2	44	36.5	76	67.9	74.7	79.1	94.16	41.09	34.41	-	64.47	74.69	75.19
Mounts	99.8	60.7	58.6	99.3	91.3	87	78.6	99.68	58.08	58.96	-	85.31	75.75	77.96
Pcb	97.5	43.1	37.5	89.2	81.7	79.6	89.5	96.13	29.74	24.58	-	77.41	78.7	85.46
Phone Battery	99.4	61.8	61.2	95.3	90.5	85.6	92.7	97.51	58.98	57.42	-	61.29	63.37	65.15
Plastic Nut	98.8	37	37.1	93.5	85.9	60.1	65.7	97.1	37.57	38.56	-	81.14	53.85	58.51
Plastic Plug	99.1	47.8	40.4	96.3	79.5	70.2	80.7	95.23	46.29	39.14	-	73.36	64.37	70.65
Porcelain Doll	99.8	45.8	45.4	99	95.2	86.2	92.7	91.65	42.4	34.37	-	63.37	52.36	50.13
Regulator	96.6	38.7	29.7	78.4	78.1	51.1	55.4	88.1	3.34	1.91	-	42.27	21.92	11.48
Rolled Strip Base	99.7	68.2	63.4	99	99	97.5	99.5	98.83	48.42	44.04	-	65.33	80.32	80.01
Sim Card Set	99.8	68.7	72.6	98.4	97.3	94	97.8	99.72	66.37	71.28	-	83.06	79.91	86.61
Switch	92.8	24.5	19.2	86.3	80.3	81.6	89	83.55	21.81	15.82	-	82.29	82.49	89.5
Tape	99.8	58.8	57.5	99.4	98.4	92.8	97.9	98.6	48.59	46.93	-	96.95	89.64	95.18
Terminalblock	99	65.2	60.7	96.7	92.8	89.9	95.9	98.53	52.16	50.18	-	61.13	71.85	68.61
Toothbrush	98	47.1	40.4	93.7	87.3	84.3	92.8	98.48	45.37	43.02	-	61.84	78.65	69.81
Toy	84.2	26	17.8	75.8	80.3	83.3	89.9	80.32	19.47	12.37	-	47.04	80.13	68.09
Toy Brick	98.9	56.5	56.9	91.2	85.9	75.6	85.2	97.73	32.03	25.41	-	54.69	59.04	43.9
Transistor	94.7	37	27.2	80.2	79.4	80.3	88.6	86.28	21.05	12.47	-	59.39	77.97	72.56
U Block	99.2	53.8	50.2	95.8	87.7	77.3	83.3	95.71	32.23	22.41	-	78.29	69.38	75.75
Usb	99.1	47.5	41.4	96.7	83.1	73.9	82.6	96.67	49.59	45.06	-	54.48	39.1	39.55
Usb Adaptor	98.8	37.8	28.4	92.5	86.9	77.5	84.3	97.63	42.81	33.58	-	80.96	74.29	80.75
Vcpill	98.3	67	65.4	88.5	84.3	74.8	82	95.45	43.35	40.93	-	52.28	51.11	43.74
Wooden Beads	98.4	47.6	44.2	89.6	79.5	75.4	86.2	95.39	19.8	13.34	-	69.82	72.57	77.64
Woodstick	99.1	63.7	66.7	96.7	92	72.7	78.9	99.57	58.02	59.74	-	78.77	54	51.17
Zipper	98	40.7	36.9	96.1	97.9	96.6	98.8	98.51	44.78	41.15	-	88.31	86.38	94.81
Average	97.9	49.4	45.7	91.9	85.8	79.5	85.8	95.39	40.51	36.73	-	70.18	68.15	68.57

Table 33. Results for MultiADS and the most competitive baseline approach, April-GAN, for each product of the MAD dataset on few-shot (k=4) anomaly detection and segmentation tasks. Both models are trained on the MVTec-AD dataset.

Baseline	MultiADS							April-GAN						
MAD	Pixel-Level				Image-Level			Pixel-Level				Image-Level		
Product	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP	AUROC	F1-max	AP	AUPRO	AUROC	F1-max	AP
Bear	91.8	16.9	11.9	82.9	71.9	93.7	94.6	91.2	13.1	8.5	79.8	64.1	93.5	92.5
Bird	91.5	9.3	4.9	76.6	64.8	94.4	92.6	90.8	7.9	4.6	74.4	66.3	94.4	93.8
Cat	94.4	8.7	4.9	86.4	57	94.5	92.3	94.1	9.2	5.6	84.5	58.4	94.5	92.6
Elephant	72.5	6.7	3.8	67.4	72.9	93.9	95.8	71.5	6.7	3.7	65.7	64.6	93.9	94
Gorilla	93.3	11.8	5.9	82.2	52.1	96.2	92.7	92.3	10.1	5.7	77.3	55.4	96.2	93.9
Mallard	86.9	14.4	6.7	67.2	62	95.6	95	86.3	15.4	8	64.6	55.7	95.6	93.8
Obesobeso	95.1	20.7	13.2	89.5	58.7	94.5	90.8	94.2	17.2	11.6	86.5	64.2	94.1	93.7
Owl	92.8	15.9	9.6	81.4	72.6	93.2	94.2	92.4	12.5	7.5	79.7	67	93	93.4
Parrot	85.7	9.2	5.1	66	66.5	92	91.7	85.2	7.2	4.4	68.5	59	91.8	89.8
Pheonix	85.7	4.4	2	73.9	52.6	94.4	90.3	85.4	4.8	2.3	73.2	53.8	94.4	90.6
Pig	95.5	13.9	10.2	86.5	61	94	93.2	95.3	14	9.5	85	62.9	94	93.9
Puppy	88.2	12.8	7.7	75.2	68.7	92.9	94.1	87.5	9.8	6.9	72.6	63.4	92.9	92.6
Sabertooth	91.7	6.4	4.7	77.6	63.8	93.2	92.9	91	5.9	4.2	74.9	60.6	93.1	91.9
Scorpion	90.7	8.7	6.2	82.7	62.1	92.9	91.8	91	8.8	6.8	81.7	65.2	92.9	93.3
Sheep	94.2	12.5	9	85.4	63.5	93.3	93.1	94.2	12.1	8.8	84.6	60.5	93.3	92.7
Swan	91	10.6	4.3	77.4	51	93.3	89.1	90.7	8.5	3.9	76.4	57.3	93.3	90.4
Turtle	91.5	12.6	7.7	77	59.6	95.2	93.7	90.9	15.4	9.4	74.2	62.6	95.2	95
Unicorn	87.6	5.1	4.1	74.3	54.6	95.7	94	87.3	5.3	4	71.3	60	95.7	95
Whale	89.5	13.3	7.4	82	58.1	94.4	92.8	89.3	16.1	9.2	80.7	67.5	94.7	94.7
Zalika	86.6	6.6	4.9	68.9	68	93.5	93.8	86	6	4.6	65.9	65.8	93.1	93.5
Average	89.8	11	6.7	78	62.1	94	92.9	89.3	10.3	6.5	76.1	61.7	94	93.1

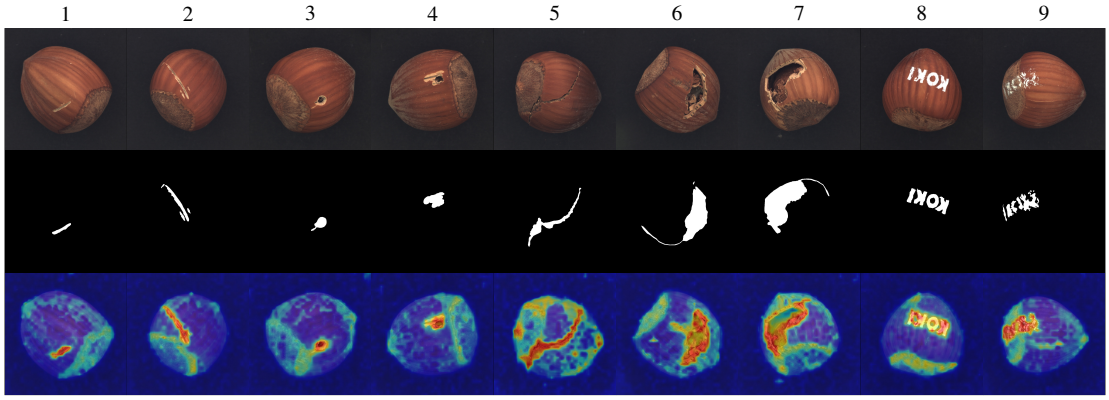


Figure 7. This visualization showcases the **hazelnut** product from the MVTec AD dataset (trained on the VisA dataset). The first row displays the input images, the second row presents the ground truth masks of anomalies, and the third row shows the predicted anomaly maps generated by the model. The model is trained on the VisA dataset and evaluated on the MVTec AD dataset using a few-shot setting with $k = 4$. As shown in the figure, our approach effectively distinguishes defect types such as **scratches** (Columns 1, 2) and **holes** (Columns 3, 4). However, for large **cracks** (Columns 6, 7), the method tends to focus on the edges while marking the interior as normal. This behavior is likely due to the patch-level features being more localized and lacking global context.

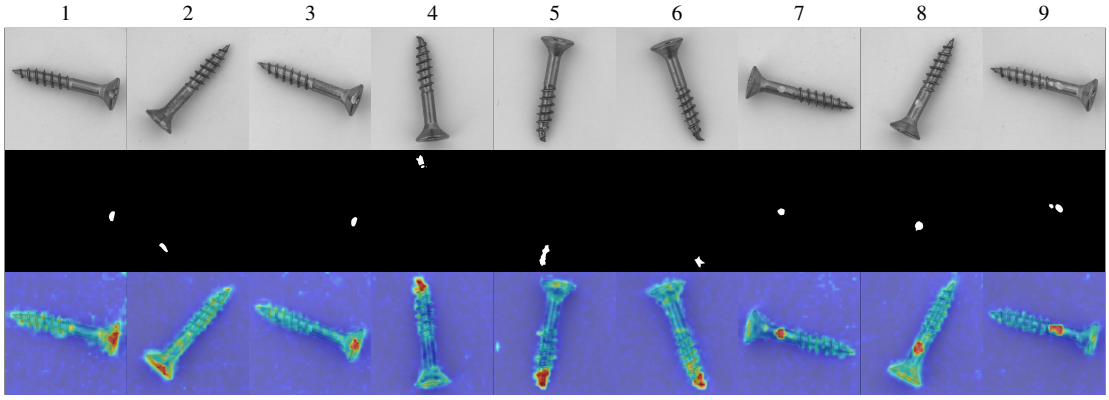


Figure 8. This visualization showcases the **screw** product from the MVTec AD dataset (trained on the VisA dataset). Our model successfully detects defects such as **scratches** (Columns 1-3, 7-9) and **bends** (Columns 4-6) in the front part. Our model also allocates some attention to the screw body.

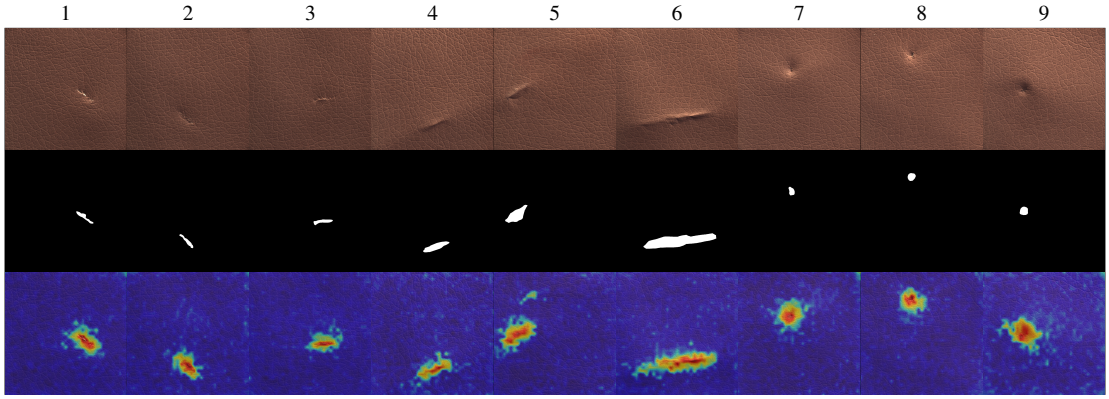


Figure 9. This visualization showcases the **leather** product from the MVTec AD dataset. Our approach can easily identify the defect of **cut** (Columns 1-3), **fold** (Columns 4-6), and **poke** (Columns 7-9).

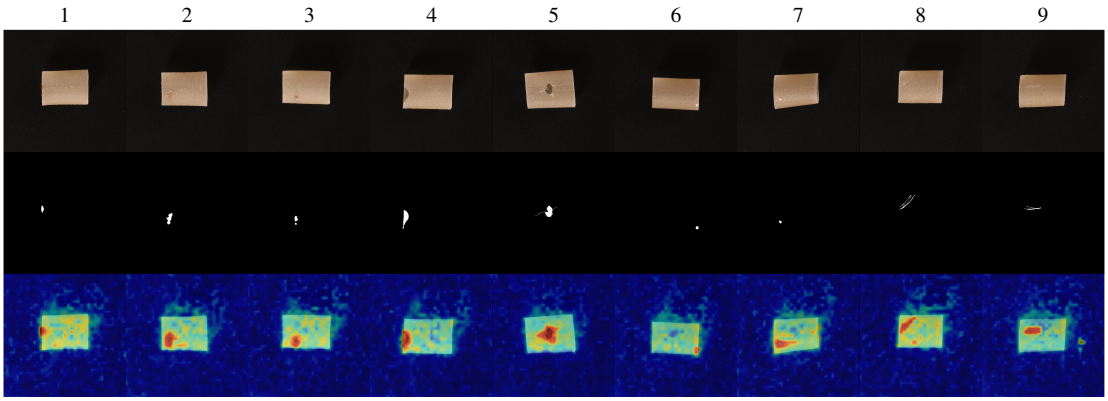


Figure 10. This visualization showcases the **pipe_fryum** product from the VisA dataset (trained on the MVTec-AD dataset). Our model can identify the defects like **color spots** (Columns 1-3), **broken** (Columns 4-5), and **scratches** (Columns 6-9).

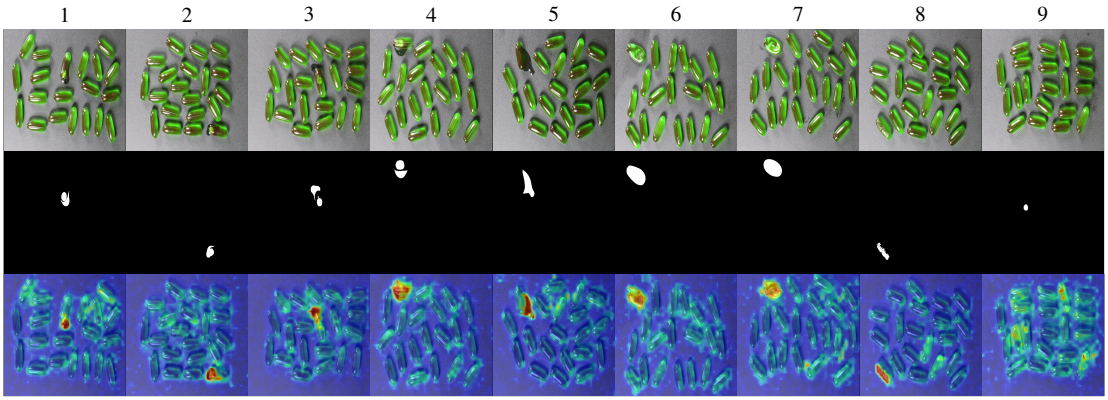


Figure 11. This visualization showcases the **capsule** product from the VisA dataset (trained on the MVTec-AD dataset). Our model effectively identifies defects such as **leakage** (Columns 1–5), **misshapes** (Columns 6–7), and **scratches** (Column 8) with clear accuracy. However, it tends to overlook **bubble** defect (Columns 1 and 9), and product highlights are occasionally misclassified as defects (Column 9).

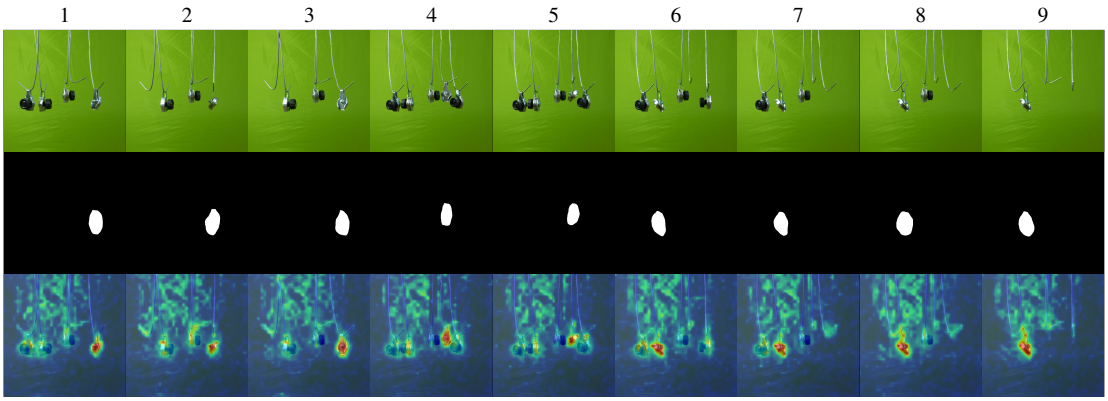


Figure 12. This visualization showcases the **connector** product from the MPDD dataset (trained on the MVTec-AD dataset). Our model effectively identifies **part-missing** defects. However, wrinkles in the green background can sometimes mislead the model, causing them to be misclassified as anomalies.

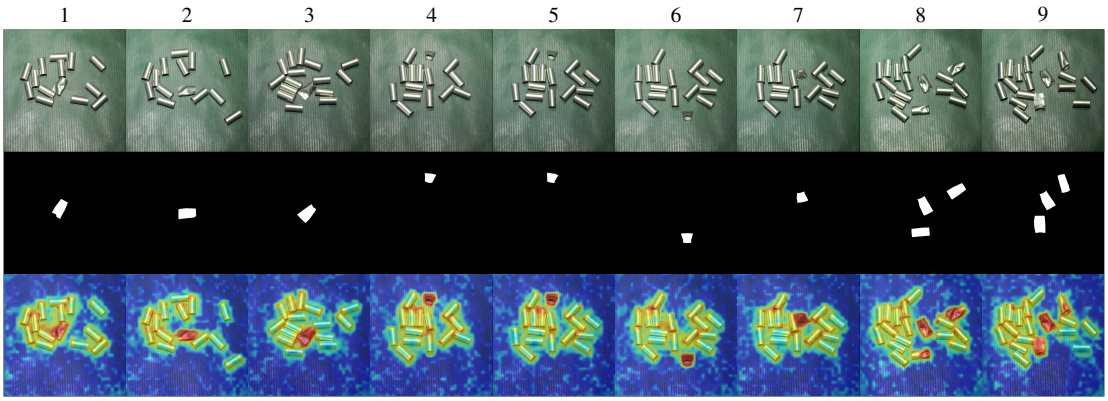


Figure 13. This visualization showcases the **tube** product from the MPDD dataset (trained on the MVTec-AD dataset). Our model successfully identifies **flattened** tubes but also introduces some noise, such as misclassifying the edges of the tubes as anomalies.

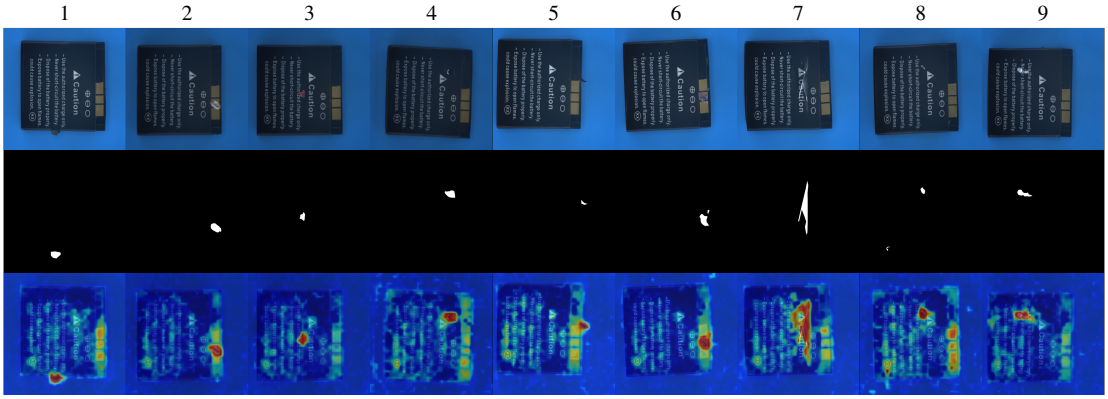


Figure 14. This visualization showcases the **phone battery** product from the Real-IAD dataset (trained on the MVTec-AD dataset). Our model successfully identifies defects like **contamination**, **scratch**, and **damage**.

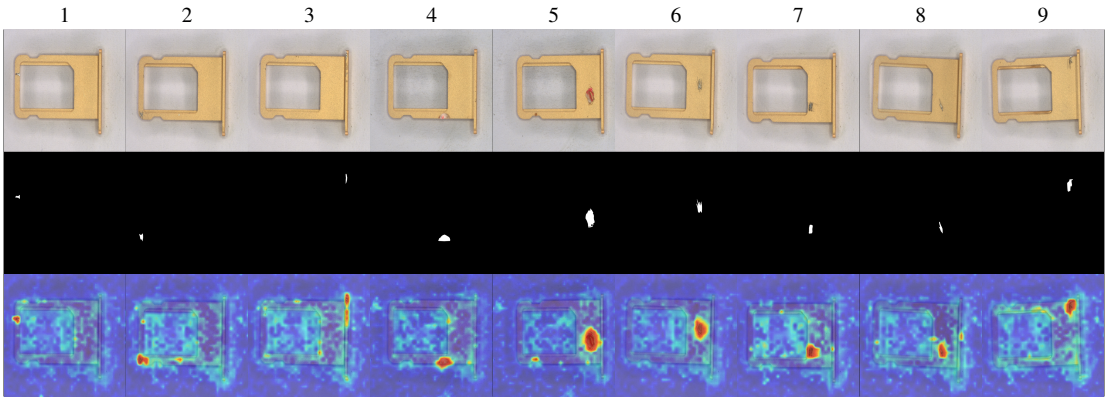


Figure 15. This visualization showcases the **sim card set** product from the Real-IAD dataset (trained on the MVTec-AD dataset). Our model successfully identifies defects like **scratch** and **damage**