AA-CLIP: Enhancing Zero-Shot Anomaly Detection via Anomaly-Aware CLIP

Supplementary Material

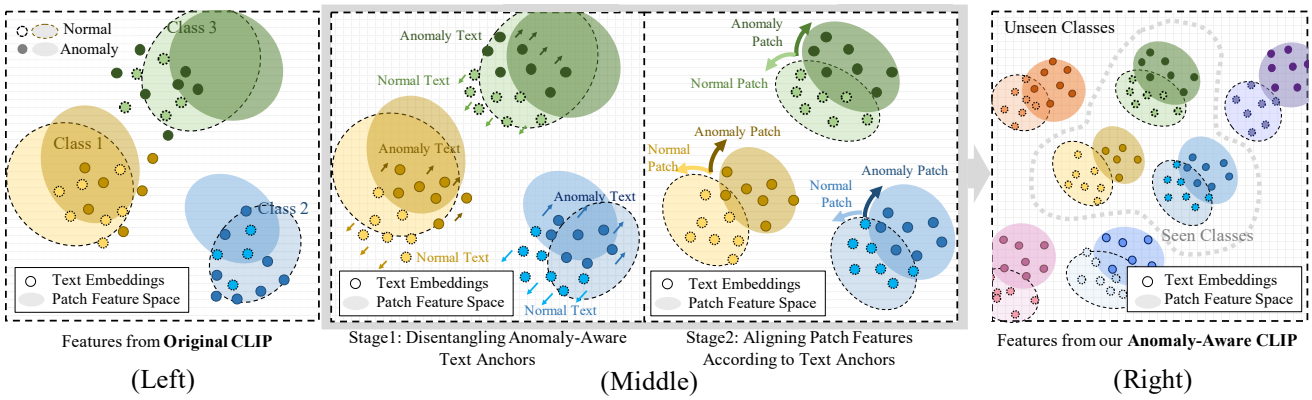

Figure 8. **(Left) CLIP's anomaly unawareness:** Category-level image-text alignment in pre-training leads to CLIP's vague distinctions in anomaly/normal semantics and inaccurate patch-text alignment. **(Middle) Our two-stage adaptation strategy:** In Stage1, anomaly and normal text features are disentangled as anchors in text space; in Stage2, patch-level visual features are trained to align to these anchors, forming Anomaly-Aware CLIP. **(Right) Generalizable anomaly awareness:** Our method enables CLIP with generalizable anomaly awareness for both known and unseen classes.

This Supplementary Material contains the following parts: 1) Additional Information about SOTA models, datasets and implementation details in Sec. A; 2) Additional experiments and further analysis about prompts, hyper-parameters, and data-efficient training in Sec. B; 3) Pseudo-Code Implementation for AA-CLIP in Sec. C; 4) Additional visualization of image feature t-SNE and anomaly maps in Sec. D.

A. Additional Information

A.1. SoTA Models

To demonstrate the superiority of AA-CLIP, we compare our methods with baseline CLIP [42] and broad recent SoTA models. The introductions are given as follows:

- **CLIP** [42], is a powerful multimodal model that learns to associate images and text through contrastive learning on large-scale datasets. By jointly training image and text encoders, CLIP can perform zero-shot classification, enabling it to understand and relate visual and textual information without task-specific fine-tuning. We use the prompts in VAND [7] to generate predictions for anomaly detection tasks.
- **WinCLIP** [19], published in CVPR 2023, enhances CLIP with a compositional ensemble of prompt templates and an efficient feature extraction method across image patches. It efficiently extracts and aggregates window/patch/image-level features to align with text, making progress in anomaly detection and segmentation tasks. The results of WinCLIP in our paper are copied from [18, 59].
- **VAND** [7] is a winning approach for the CVPR 2023 Zero/Few-shot Track of the Visual Anomaly and Novelty Detection (VAND) 2023 Challenge. It enhances CLIP with additional linear layers to map image features into a shared embedding space for anomaly map generation. The results of VAND in our paper is inferred from their official weights.
- **MVFA-AD** [18] is a work adapting CLIP to AD tasks, published in CVPR 2024. It proposes a multi-level adaptation framework to progressively refine image features. They mainly focus on medical domain in their original work. We re-train the model using the original code and original configuration to validate its generalization ability, except using VisA as training dataset.
- **AnomalyCLIP** [59] is published in ICLR 2024, which is the first work targeting prompt-learning techniques to improve CLIP. It learns object-agnostic text prompts that capture general indicators of normality and abnormality, allowing the model to detect anomalies without being tied to specific object semantics. The results in our paper is inferred from the official weight.
- **AdaCLIP** [6] published in ECCV 2024. It incorporates static and dynamic prompts allowing for shared and real-time adaptation, demonstrating strong zero-shot performance and generalization across diverse datasets. Their original implementation incorporates training in both industrial and medical datasets. We re-train the model with identical setting using VisA dataset.

A.2. Datasets

We conduct extensive experiments on ten public datasets from industrial and medical domains, covering five modalities. For each dataset, we use only the test set. Relevant details of the datasets are presented in Tab. 4.

Dataset	Domain	Modalities	Classes	Number of Samples	
				Normal	Anomaly
MVTec-AD	Industrial	Photography	15	467	1258
MPDD	Industrial	Photography	6	176	282
BTAD	Industrial	Photography	3	451	290
Brain MRI	Medical	MRI	1	640	1013
Liver CT	Medical	CT	1	833	660
Retina OCT	Medical	OCT	1	1041	764
ColonDB	Medical	Endoscopy	1	0	380
ClinicDB	Medical	Endoscopy	1	0	612
Kvasir	Medical	Endoscopy	1	0	1000
CVC-300	Medical	Endoscopy	1	0	60

Table 4. Key Information of The Datasets Used

A.3. Additional Implementation Details

During training, all images are resized to 518×518 pixels. Data augmentation techniques—including color jitter, random rotation, random affine transformation, random horizontal flip, and random vertical flip—are applied with a probability of 0.5. Additionally, the default normalization of OpenCLIP is applied to all datasets during both training and inference. The hyper-parameters for Adam optimizer are set to $\beta_1 = 0.5$, $\beta_2 = 0.999$, following previous work [18]. For the first stage, the batch size is set to 16, while for the second stage, the batch size is 2.

A.4. Prompts

The prompt templates and descriptors we use are listed in Tab. 5. During both training and inference, "[CLS]" is replaced with the class description, and normal or anomaly descriptors are inserted into the templates. Normal and anomaly anchors are generated by averaging the embeddings of different prompts.

State	Prompt
Prompt Template	{} a photo of a {}
Normal Descriptors	[CLS] a [CLS] the [CLS]
Anomaly Descriptors	damaged [CLS] broken [CLS] [CLS] with flaw [CLS] with defect [CLS] with damage

Table 5. Templates of Prompts Used

Considering the setting of AD aims to identify anomalies without prior knowledge of the specific anomaly types, we use only general adjectives to represent anomaly semantics, as shown in (Bottom). To describe a class, we utilize GPT-4 to automatically generate descriptions. Notably, only normal samples are input to GPT-4 to ensure proper generation, as shown in Fig. 9.

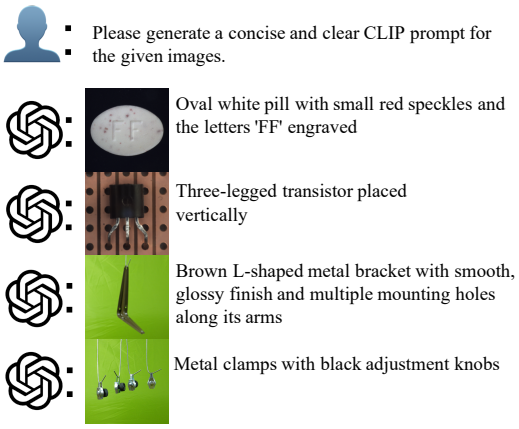

Figure 9. Examples of GPT-4 Generated Class Discription.

B. Additional Experiments

B.1. Prompts

B.1.1. Templates

To demonstrate the robustness of our model across various prompt templates, we present additional experimental results in Tab. 6. We obtain inference results from the full-shot trained model using both seen and unseen prompt templates, testing on the BTAD and Retina OCT datasets. The results with novel prompt templates indicate that the effectiveness in differentiating anomalies is not dependent on the specific prompt templates used.

Prompt		BTAD		Retina OCT	
		Pixel	Image	Pixel	Image
Seen	{}.	97.0	94.7	95.5	82.7
	a photo of {}.	97.4	94.8	95.6	82.7
Unseen	a picture of {}.	97.3	94.6	95.4	82.3
	this is a photo of {}.	97.3	94.6	95.7	82.4
	there is a {}.	97.2	95.1	95.6	82.3
	a {} in the scene.	96.7	95.4	95.6	82.1
	a close-up photo of {}.	97.3	94.7	95.6	82.4

Table 6. AUROC Results of Different Prompt Templates Used

B.1.2. Anomaly Semantics

To demonstrate that our model successfully learns generalizable anomalous semantics, we present additional experimental results using different anomaly descriptors in Tab. 7, including seen and unseen ones. We obtain inference results from the full-shot trained model using both seen and unseen

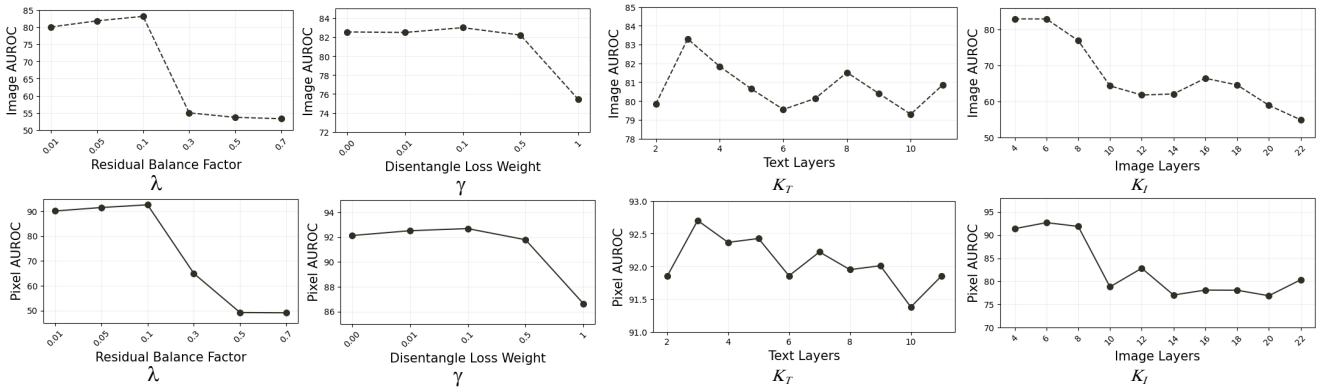

Figure 10. Ablation Study of Hyper-parameters

anomaly descriptors on the MPDD and Retina OCT datasets as examples. The results with novel anomaly descriptors indicate that the effectiveness in differentiating anomalies is not dependent on the specific adjectives used to describe anomaly in the prompts, further validating that our model successfully equips CLIP with anomaly-awareness.

Prompt		MPDD		Retina OCT	
		Pixel	Image	Pixel	Image
Seen	damaged [CLS]	96.6	75.0	95.4	81.8
	broken [CLS]	96.8	75.1	95.4	82.6
	[CLS] with flaw	96.7	75.6	94.6	82.8
	[CLS] with defect	96.6	75.3	95.6	82.6
	[CLS] with damage	96.5	75.4	95.4	82.6
Unseen	corrupted [CLS]	94.3	72.9	93.3	82.5
	defective [CLS]	96.4	75.1	95.5	82.9
	flawed [CLS]	96.6	74.4	95.5	82.6
	impaired [CLS]	96.7	75.5	94.9	80.9
	[CLS] with defective parts	96.1	73.0	95.2	81.7
	[CLS] with damaged parts	97.1	73.0	95.4	81.5

Table 7. AUROC Results of Different Anomaly Descriptors Used

B.2. Additional Ablations for Hyper-parameters

We provide additional results of different hyper-parameter settings in this section, as shown in Fig. 10. The hyper-parameters include:

- λ controls the residual fusing ratio,
- γ controls the weight of Disentangle Loss L_{dis},
- K_T/K_I controls layers of inserted adapters in text/visual encoder.

A large residual ratio λ means that the output contains less information from CLIP and more information from linear adapter, allowing more flexible adaptation and leading to improvement when $\lambda \leq 0.1$. However, due to the modeling limitation and catastrophic forgetting, zero-shot performance drops dramatically after $\lambda \geq 0.3$.

Since L_{dis} is a normalization loss without information flowing from training data, a large γ can be detrimental to the learning process because of semantic disturbance to CLIP's feature space, leading to performance drop.

For K_T/K_I, inserting adapters in shallow layers prevents drastic semantic changes caused by inserting adapters in deep layers and maintains training stability. Large K_T/K_I damages overall performance significantly because of higher possibility of catastrophic forgetting with more trainable parameters. Since it contains more semantically meaningful information in text space, the performance fluctuation for text encoder is more obvious compared to image encoder.

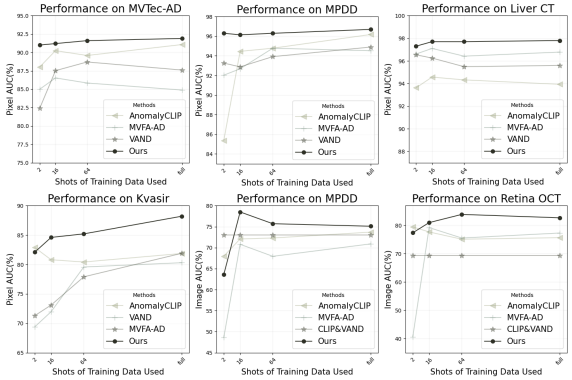

Figure 11. Additional Results under Different Data Levels

B.3. Comparison under Various Data Levels

We re-implement SOTA methods for comparison under different settings: 2-shot per class, 16-shot per class, 64-shot per class, and full-shot, presented in line-figure format. In addition to the BTAD results shown in our main paper, additional figures are displayed in Fig. 11. Our method demonstrates relatively stable superiority across all methods, maintaining a leading position in most scenarios. These results prove that our training strategy has high fitting efficiency.

Despite notable domain gaps among test datasets, which can cause some variability in results, our model's performance generally improves as training data increases, with minor fluctuations. We provide an explanation as follows: with limited data, the adapted image encoder initially leverages textual cues, effectively identifying anomalies while occasionally leading to false positives. As more data becomes available, the model captures richer and more balanced features, further enhancing accuracy. However, with excessive data, the model may start to capture domain-specific details over broader abnormality semantics, reducing generalization to unseen anomalies. This trend aligns with our visualizations and is consistent across our implementations of other methods, leaving space for further investigation on stable adaptation.

C. Pseudocode Implementation for AA-CLIP

To clearly demonstrate AA-CLIP model, we present our model definition using PyTorch-style pseudocode, as shown Fig. 12.

```
class AdaptedCLIP(nn.Module):
    # forward function of text encoder with ResidualAdapters
    # x: bs,2,d
    def forward_text(x):
        layer = 0
        while layer < K:
        # layers with adapters inserted in
            x = CLIP_text_encoder[layer](x)
            x_adapted = ResidualAdapter(x)
            # residual adaptation
            x = λ * x+(1-λ) * x_adapted
            layer += 1
        while layer < CLIP Layers:
        # layers without adapters inserted in
            x = CLIP_text_encoder[layer](x)
            layer += 1
        x=Proj(x)
        return x

    # forward function of image encoder with ResidualAdapters
    # x: bs,N,d
    def forward_image(x):
        layer = 0
        tokens=[]
        while layer < K:
        # layers with adapters inserted in
            x = CLIP_image_encoder[layer](x)
            x_adapted = ResidualAdapter(x)
            x = λ * x+(1-λ) * x_adapted
            if layer in Output Layers:
                tokens.append(x) # intermediate output
            layer += 1
        while layer < CLIP Layers:
        # layers without adapters inserted in
            x = CLIP_image_encoder[layer](x)
            if layer in Output Layers:
                tokens.append(x) # intermediate output
            layer += 1
        x=[Proj_i(x) for i in range(len(Output Layers))]
        return x
```

Figure 12. Pseudocode Implementation of AA-CLIP

D. Additional Visualization

D.1. Image t-SNE Visualization

In our main paper, we present t-SNE visualizations for text embeddings of both seen and unseen classes. Additionally, we include further visualizations of image features for seen classes in Fig. 15 and for unseen classes in Fig. 16. The results for original CLIP and AA-CLIP are from same t-SNE optimization iterations. We keep the number of normal and anomaly patches the same for clear visualization. The results prove that our model succeed in equipping CLIP with generalizable anomaly discrimination ability.

D.2. Anomaly Maps Visualization

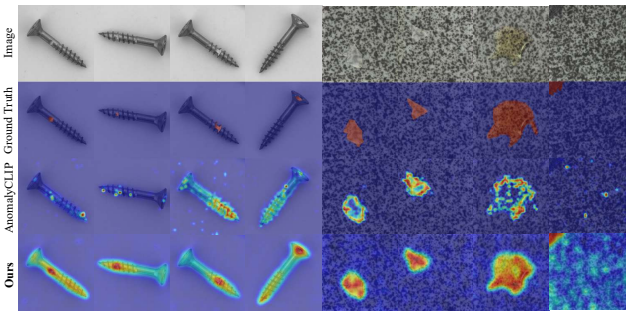

Figure 13. Additional Visualization Examples of AnomalyCLIP and AA-CLIP.

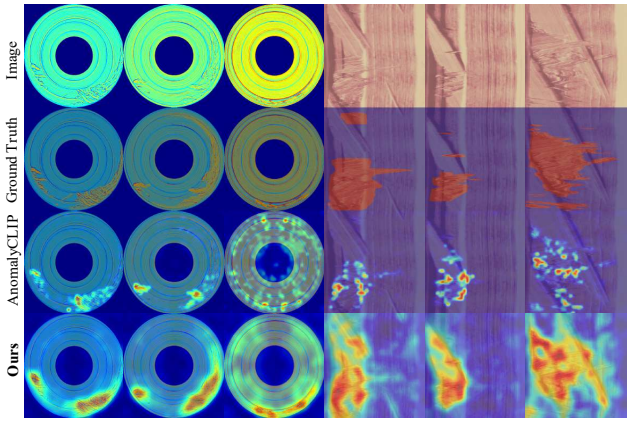

Figure 14. Additional Visualization Examples of AnomalyCLIP and AA-CLIP.

We provide additional visualization examples of AA-CLIP to illustrate the effectiveness of our method. We show more comparative results of recent SOTA AnomalyCLIP and our methods in Figs. 13 and 14. Despite AnomalyCLIP can identify anomalous regions, our model's predictions have less false-positive predictions and localizes anomalous regions more precisely.

We provide more examples of our AA-CLIP's prediction, as shown in Figs. 17 to 28. For each figure, the first row shows the input, with anomaly regions highlighted in red in the second row. The final row displays the segmentation results generated by AA-CLIP. Our model shows precise localization results in general.

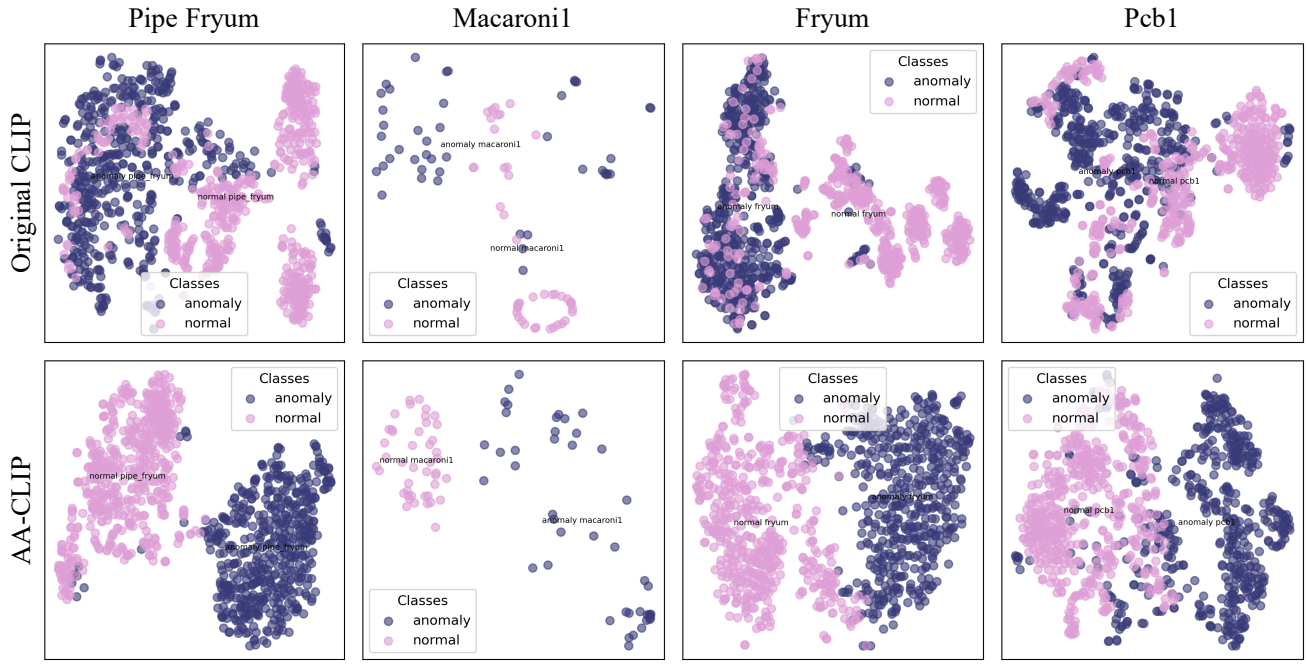

Figure 15. Additional t-SNE Visualization of Visual Features of Seen Classes

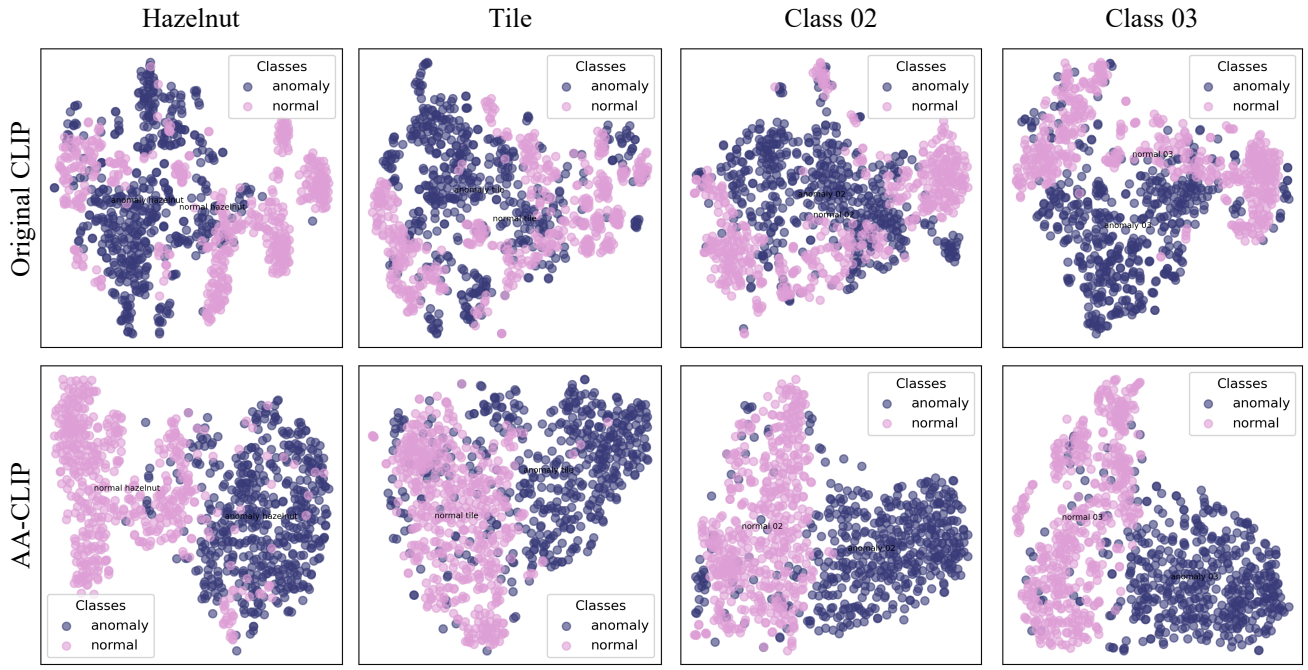

Figure 16. Additional t-SNE Visualization of Visual Features of Unseen Classes

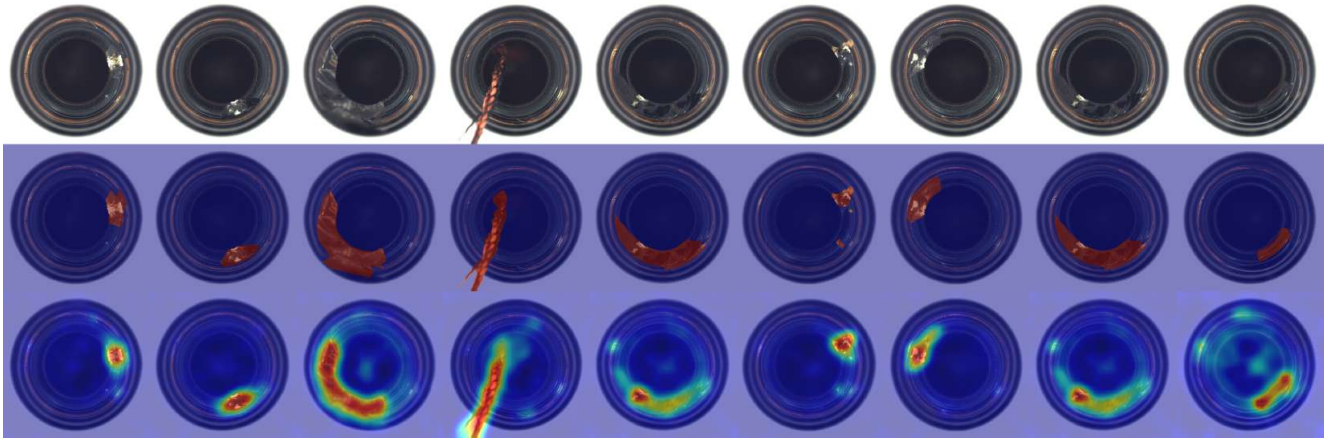

Figure 17. Visualization Examples of Class Bottle in MVTec-AD dataset.

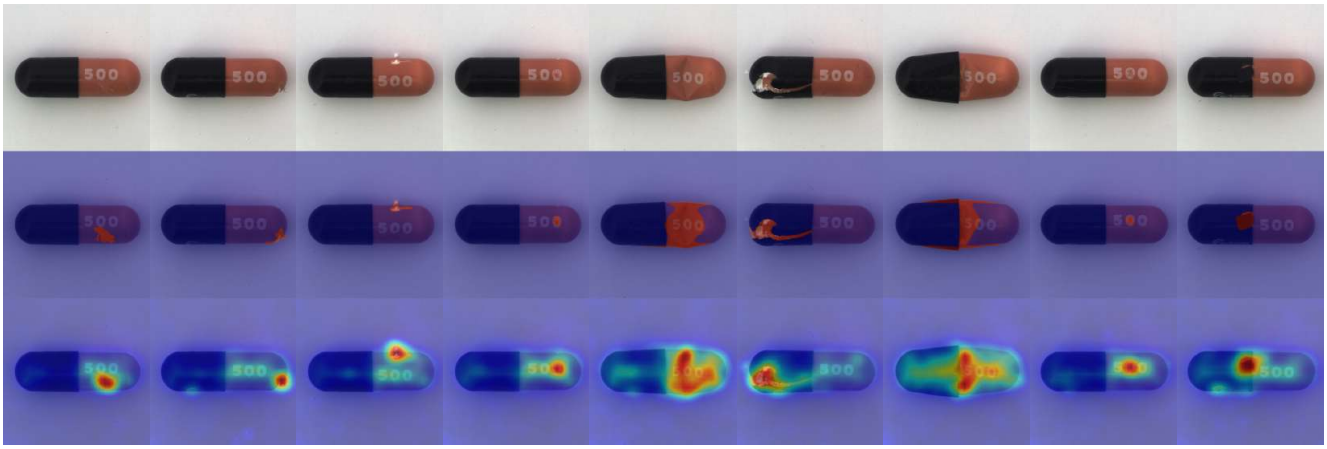

Figure 18. Visualization Examples of Class Capsule in MVTec-AD dataset.

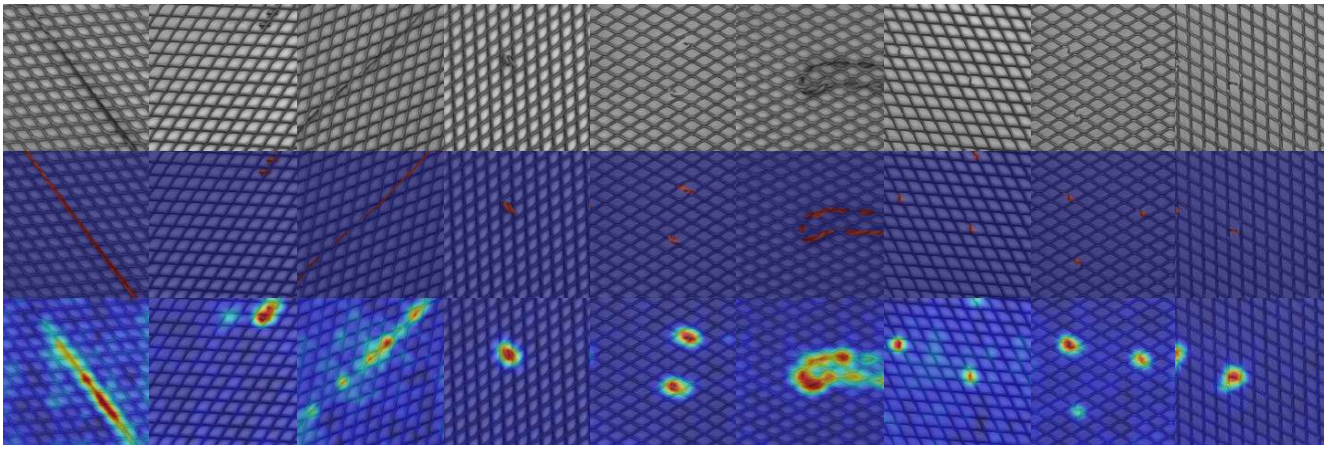

Figure 19. Visualization Examples of Class Grid in MVTec-AD dataset.

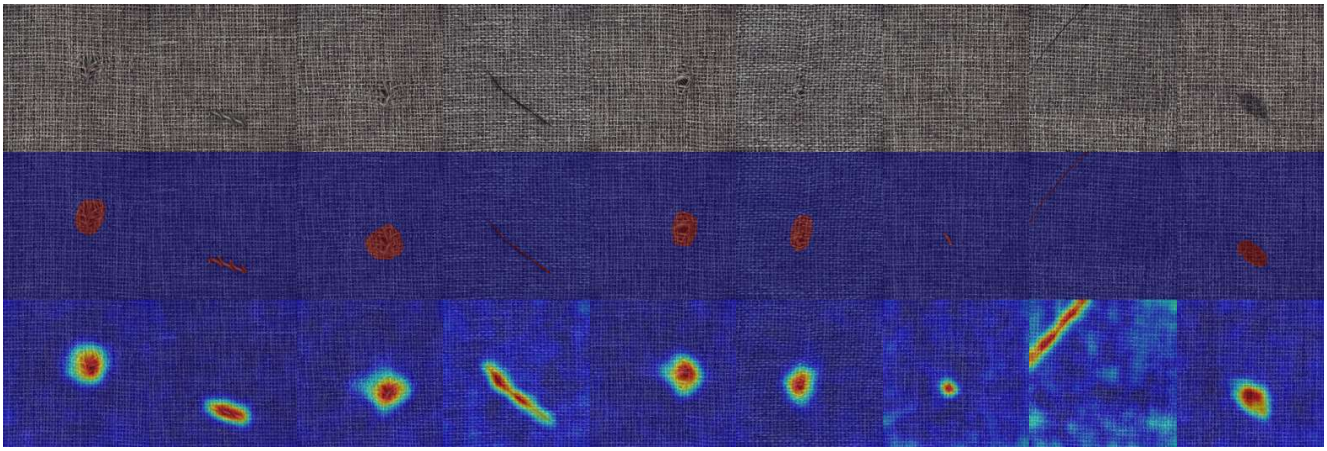

Figure 20. Visualization Examples of Class Carpet in MVTec-AD dataset.

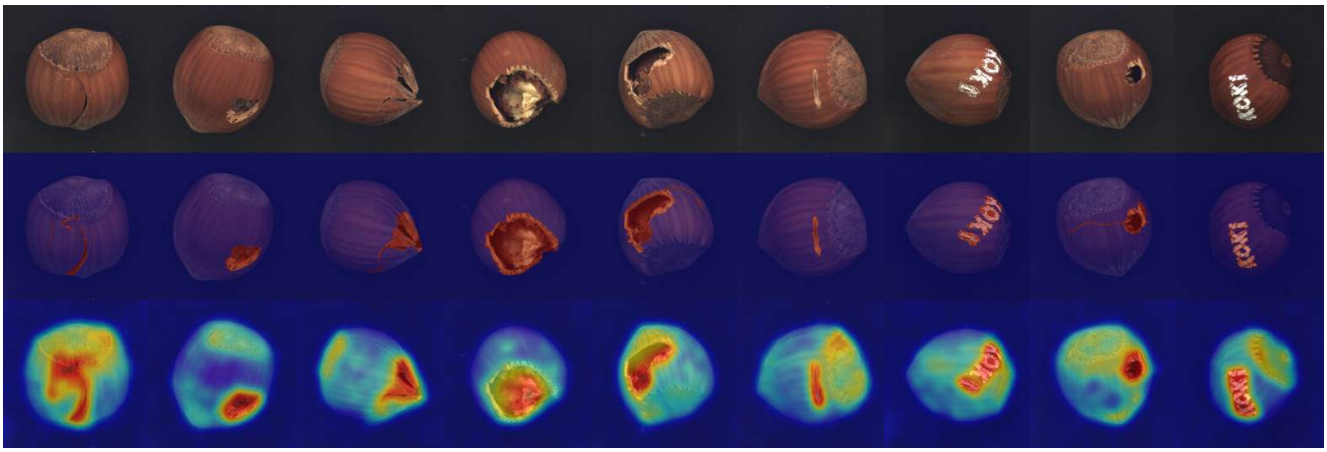

Figure 21. Visualization Examples of Class Hazelnut in MVTec-AD dataset.

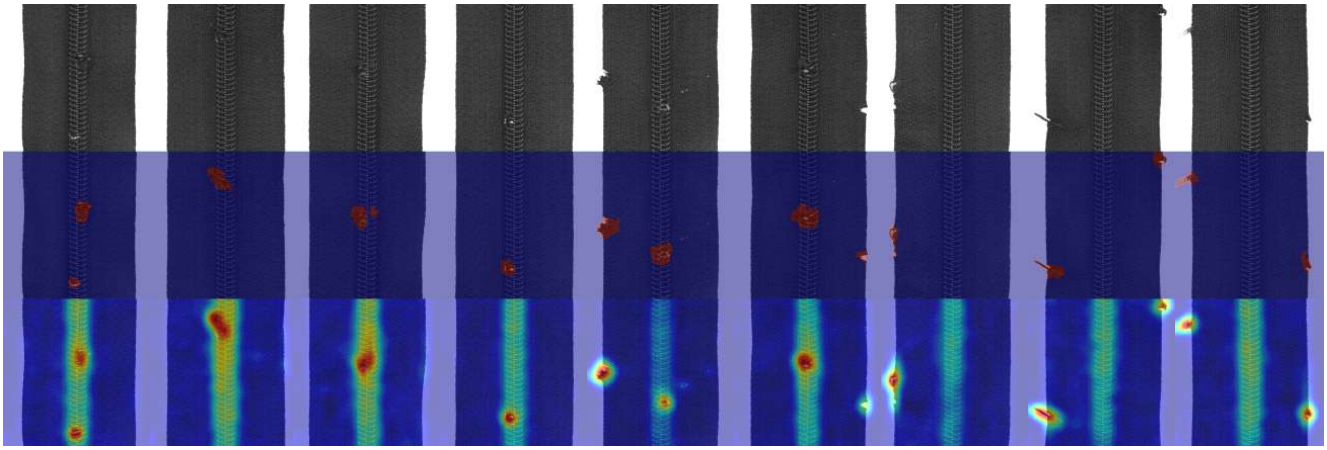

Figure 22. Visualization Examples of Class Zipper in MVTec-AD dataset.

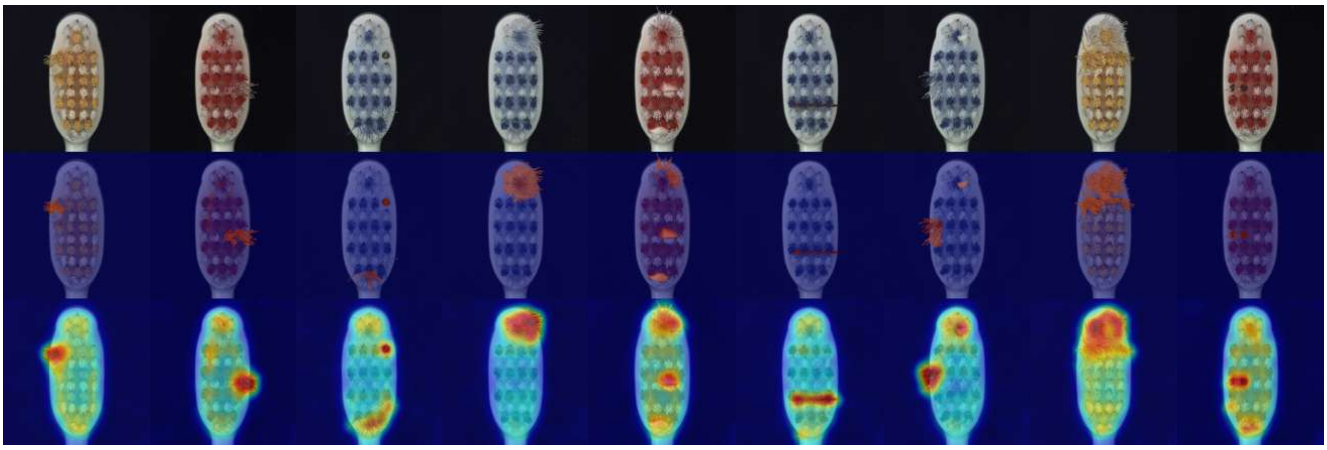

Figure 23. Visualization Examples of Class Toothbrush in MVTec-AD dataset.

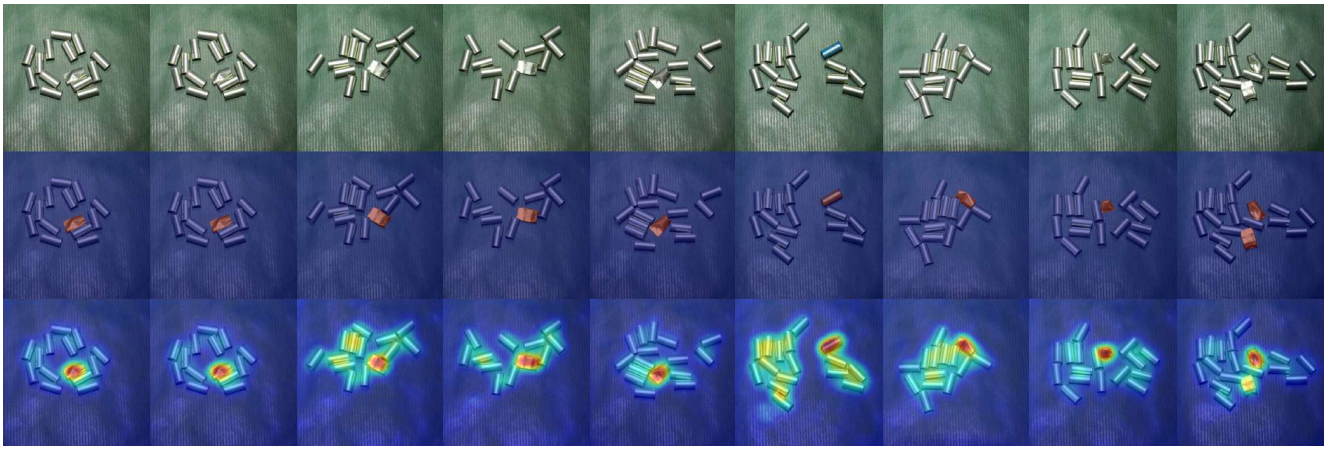

Figure 24. Visualization Examples of Class Tubes in MPDD dataset.

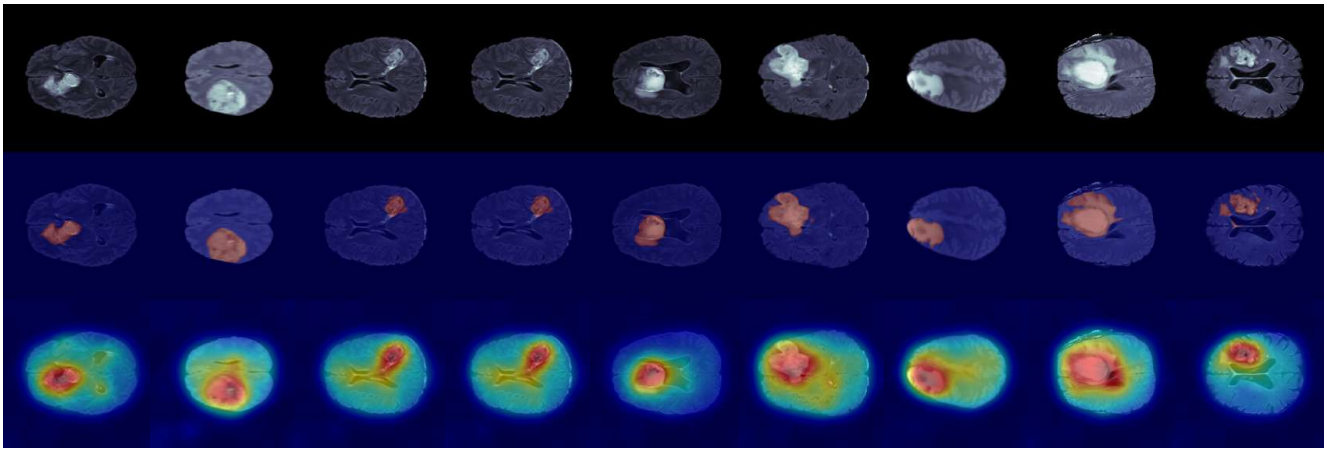

Figure 25. Visualization Examples of Brain MRI dataset.

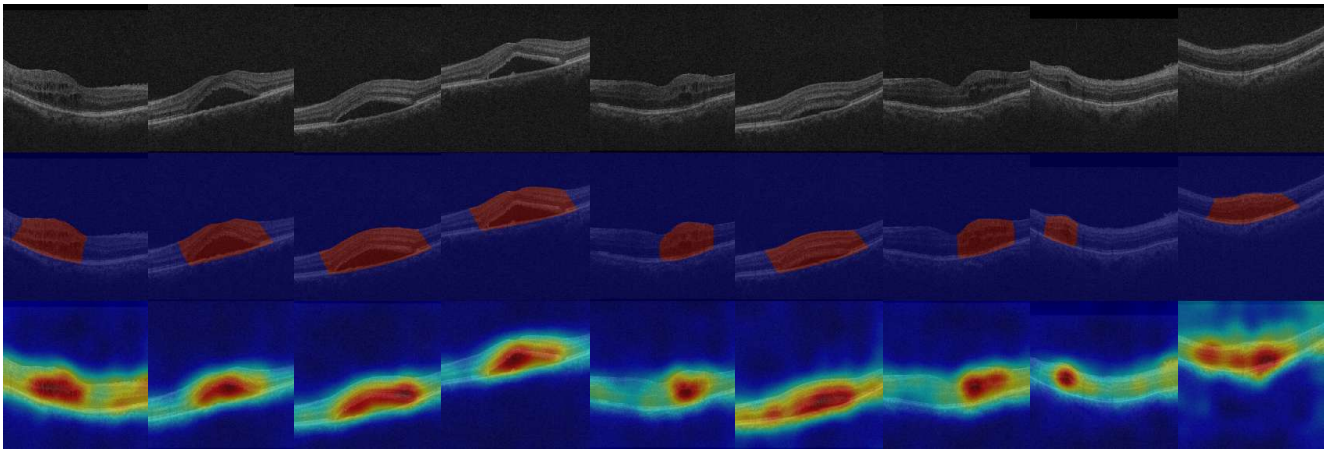

Figure 26. Visualization Examples of Retina OCT dataset.

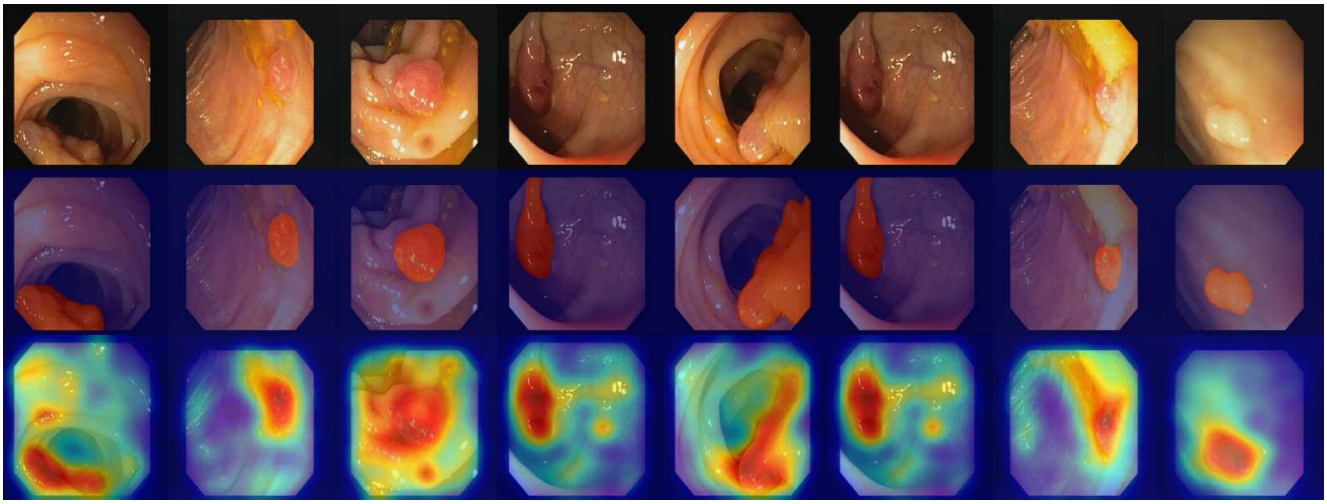

Figure 27. Visualization Examples of endoscopic photos in ClinicDB dataset.

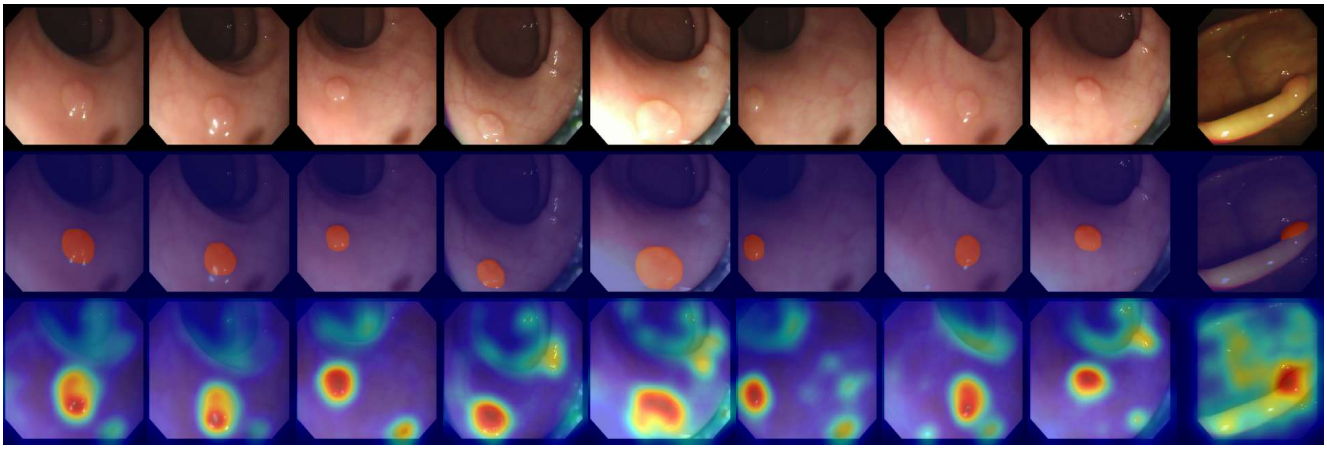

Figure 28. Visualization Examples of endoscopic photos in CVC-300 dataset.