# ObjectCore – Efficient Few-shot Logical Anomaly Detection using Object Representations

## Supplementary Material

This supplementary material includes additional information and visualisations. More specifically, we first discuss the limitations of the proposed method and suggest improvements that we left for future work. Then, we will analyse the influence of different parameters on object detection and anomaly detection performance. Afterwards, we analyse different aspects of the proposed method. More specifically, we look into the effect of rotating objects, the effects of feature smoothing on the detection of smaller defects and the impact of missing pseudo-detection on the final anomaly score. Then we visualise pseudo-detections generated with different hyperparameters. Ultimately, we add more qualitative examples.

## A. Limitations

ObjectCore requires all the different representations of overall normality in the support set for the best performance. For example, in the juice bottle category, the bottle can be filled with either cherry, banana or lemon juice. If one of these images is missing from the support set, ObjectCore will mark it as an anomaly because it was not seen before. If we ensure all of the different normal representations are in the support set for ObjectCore, the performance in I-AUROC rises by 1.6 percentage points with $k = 4$.

The second limitation of ObjectCore is the reliance on GroundingDINO for pseudo-label extraction. If GroundingDINO [5] fails to detect an important object, the method will not perform as well as it could. This can easily be fixed, of course, by providing hand-made bounding boxes. The pseudo-label extraction procedure also applies only to GroundingDINO, and the parameters and the text prompt must be adjusted for different open-world object detectors.

## B. Additional ablation study results

In this section, we present some additional experiments validating ObjectCore. First, we validate the object detection performance following the experimental setup described in the main paper. And finally, we look at the impact of those parameters on the downstream performance of anomaly detection.

### B.1. Object Detection Performance

**Open-world object detector thresholds** The object detector depends on a threshold to determine what is output as a detection and what is not. By default, 0.25 is used as the threshold. To evaluate the importance of this parameter, we have two other values: 0.2 and 0.3. The results indicate that 0.25 is the best choice. Nevertheless, the results indicate that the model is still quite robust in terms of the value of the threshold.

| Group | Condition | MVTec LOCO | |
|---|---|---|---|
| | | MAE | AP |
| *GroundingDINO* | Threshold = 0.2 | 1.2 | 71.5 |
| | Threshold = 0.3 | 1.9 | 65.3 |
| | with $\mathcal{L}_{bbox}$ | 3.4 | 44.5 |
| *ObjectCore* | Threshold = 0.25, w/o $\mathcal{L}_{bbox}$ | 0.8 | 71.6 |

Table 1. Ablation study results for $k = 4$ concerning object detection and localization results. For MAE, lower is better, and for AP, higher is better.

| Group | Condition | MVTec LOCO | | | |
|---|---|---|---|---|---|
| | | I-AUROC | I-$F_1$-Max | P-AUROC | P-$F_1$-Max |
| *Grounding DINO* | Threshold = 0.2 | -1.4 | -1.1 | -0.8 | -0.8 |
| | Threshold = 0.3 | -0.4 | -0.9 | -1.8 | -1.9 |
| *ObjectCore* | | 80.8 | 82.8 | 84.4 | 31.9 |

Table 2. Ablation study results for $k = 4$. In each row the difference to the actual model is shown. The highest discrepancy for each experiment group is marked in blue.

**Training Losses** The object detector is fine-tuned using only the $\mathcal{L}_{cls}$ loss. Original GroundingDINO also uses the $\mathcal{L}_{bbox}$, which, in our case, penalises the model for detecting objects not contained in the pseudo-detections. We have evaluated the model trained without this loss in terms of object detection performance. The results clearly show that the model performs worse with the addition of this loss.

### B.2. Anomaly Detection Performance

**Feature Smoothing** The extracted features are first smoothed with a mean kernel (a 5x5 kernel) to infuse the features with patch information. We have used several different kernel sizes to verify the robustness of ObjectCore towards this parameter. The results can be seen in Figure 1 and Figure 2. It can be discerned that ObjectCore is indeed robust in the choice of this parameter.

**Anomaly Map Smoothing** The anomaly map $A$ is smoothed using a Gaussian Kernel (with $\sigma = 4$) before calculating the final anomaly score. To show the robustness of this choice, we experimentally verified how different values of $\sigma$. The results are depicted in Figure 1. It can be discerned that the proposed model is robust to this choice. Even more, the

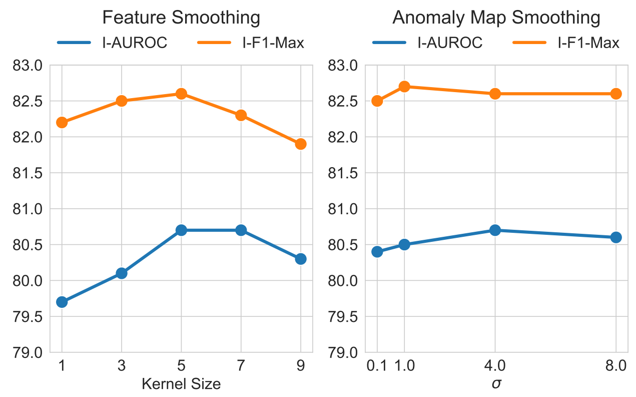

Figure 1. Anomaly Detection Performance for different kernel sizes for feature and anomaly map smoothing.

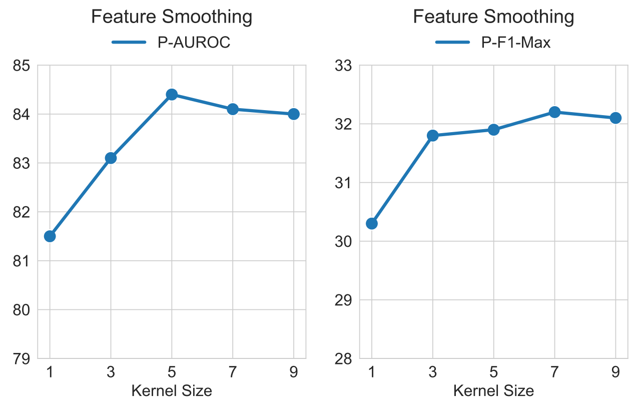

Figure 2. Anomaly Localisation Performance for different kernel sizes for feature map smoothing.

| Setup | Method | MVTec LOCO | | CAD-SD | |
|---|---|---|---|---|---|
| | | I-AUROC | I-$F_1$-max | I-AUROC | I-$F_1$-max |
| 1 + B.B. | ObjectCore[†] | 74.0±0.6 | 80.7±0.3 | 91.4±2.2 | 83.7±3.3 |
| 2 + B.B. | ObjectCore[†] | 77.9±0.3 | 81.8±0.3 | 95.1±0.8 | 87.1±2.1 |
| 4 + B.B. | ObjectCore[†] | **83.6**±0.3 | **83.0**±0.6 | **97.7**±1.1 | **91.2**±2.0 |

Table 3. Anomaly Detection on MVTec LOCO [2] and CAD-SD [3] Datasets for methods with hand-made information. ObjectCore[†] uses hand-labelled boxes during open-world detector finetuning.

results stay relatively the same no matter the choice.

**Open-world object detector thresholds** In the previous section, we evaluated the importance of the object detector performance in terms of detection performance. We have also evaluated its effect on downstream anomaly detection. The results once again show that the model is robust to this choice, as the model still achieves state-of-the-art results over other methods.

| Dataset | Category | I-AUROC | I-$F_1$-Max | P-AUROC | P-$F_1$-Max |
|---|---|---|---|---|---|
| MVTec LOCO | Screw Bag | 72.3 | 79.3 | 74.8 | 14.0 |
| CAD-SD | Screw | 96.5 | 90.1 | - | - |
| MVTec AD | Capsule | 94.9 | 96.4 | 98.7 | 50.8 |
| MVTec AD | Hazelnut | 100 | 100 | 99.4 | 75.6 |
| MVTec AD | Metal Nut | 99.9 | 99.8 | 96.1 | 74.6 |
| MVTec AD | Screw | 74.6 | 87.8 | 97.7 | 32.5 |
| VisA | Capsules | 92.2 | 88.4 | 98.3 | 44.0 |
| VisA | Fryum | 93.7 | 92.1 | 92.9 | 41.6 |
| VisA | Macaroni 2 | 68.9 | 71.9 | 97.8 | 15.2 |

Table 4. Detection performance for rolling and rotating objects

## C. Adding category-specific information

We also trained ObjectCore with bounding boxes we hand-labelled to add category-specific information. The results are shown in Table 3. We denoted the model trained with hand-made labels as ObjectCore[†] for clarity. ObjectCore[†] outperforms the model trained with pseudo-detections, showcasing the possible improvement of having additional category-specific information.

## D. Rolling and Rotating Object Analysis

Labels on objects that are rolling or rotating can be relatively unique (e.g. Capsule in MVTec AD). To investigate whether ObjectCore performs worse, we compiled the mean anomaly detection performance for such categories. The results can be seen in Table 4. The results suggest that ObjectCore performs well with categories with small amounts of rotation (e.g. Hazelnut, Capsule, Fryum), but can lead to deterioration in performance in categories with significant amounts of rotation (e.g. Macaroni 2, Screw, Screw Bag). The results suggest that rotation can cause problems. We have left these as possible places to improve in the future.

## E. Effects of Feature Smoothing on Small Defect Detection

Excessive feature smoothing can lead to the missing detection of smaller defects. To investigate this, we have measured how many of the smaller anomalies have been missed. More specifically, we checked how many anomalous images with defects spanning fewer than a certain percentage of pixels produced an anomaly score below the threshold defined by the I-$F_1$-Max and how many non-anomalous images produced a score above it. The results can be seen in Table 5. They show that feature smoothing has little effect on the downstream detection. Even more, it shows that our method is able to perfectly classify non-anomalous images.

| **Anomaly Size** | **Setup** | TP | FP | TN | FN |
|---|---|---|---|---|---|
| $< 1\%$ | w/o smoothing | 415 | 10 | 575 | 0 |
| | w/ smoothing | 408 | 17 | 575 | 0 |
| $< 2\%$ | w/o smoothing | 534 | 15 | 575 | 0 |
| | w/ smoothing | 529 | 20 | 575 | 0 |
| $< 5\%$ | w/o smoothing | 614 | 16 | 575 | 0 |
| | w/ smoothing | 607 | 23 | 575 | 0 |
| $< 10\%$ | w/o smoothing | 755 | 18 | 575 | 0 |
| | w/ smoothing | 742 | 31 | 575 | 0 |

Table 5. Small defect detection results on MVTec LOCO [2] for $k = 4$. The model misdetects a bit more anomalies when feature smoothing is used.

| **Setup** | **Missing perc.** | MVTec LOCO | | | |
|---|---|---|---|---|---|
| | | I-AUROC | I-$F_1$-Max | P-AUROC | P-$F_1$-Max |
| *1* | 0% | 72.5 | 79.8 | 76.3 | 28.8 |
| | 10% | 69.5 (-3.5) | 79.2 (-0.5) | 76.0 (-0.3) | 28.0 (-0.8) |
| | 25% | 68.8 (-4.2) | 79.4 (-0.4) | 75.8 (-0.5) | 26.9 (-1.9) |
| | 50% | 68.7 (-4.3) | 79.0 (-0.8) | 75.0 (-1.3) | 25.7 (-3.1) |
| *2* | 0% | 74.6 | 80.5 | 81.4 | 29.3 |
| | 10% | 73.8 (-0.8) | 80.2 (-0.3) | 81.2 (-0.2) | 29.1 (-0.2) |
| | 25% | 71.5 (-3.1) | 80.2 (-0.3) | 81.2 (-0.2) | 28.8 (-0.5) |
| | 50% | 70.5 (-4.1) | 79.9 (-0.6) | 80.5 (-0.9) | 27.4 (-1.9) |
| *4* | 0% | 80.8 | 82.8 | 84.4 | 31.9 |
| | 10% | 80.5 (-0.3) | 82.7 (-0.1) | 83.8 (-0.6) | 31.3 (-0.6) |
| | 25% | 78.5 (-2.3) | 82.0 (-0.8) | 83.0 (-1.4) | 30.8 (-1.1) |
| | 50% | 76.8 (-4.0) | 80.6 (-2.2) | 83.1 (-1.3) | 29.6 (-2.3) |

Table 6. Impact of missing pseudo-detections for downstream performance. The results show the performance at different rates. The difference to the baseline performance is marked in gray.

## F. Impact of Missing Objects in Pseudo-Detections

To quantitatively measure the impact of missing objects, we randomly discarded some of the pseudo-detections. More specifically, we discarded 10%, 25% and 50% of the boxes. The results can be seen in Table 6. The results show that with increasing the number of available images, the effect of missing pseudo-detections decreases. Interestingly, when 50% of the boxes are missing, I-AUROC almost always deteriorates for about 4 percentage points. The results still show that detecting a bit fewer objects is not a big problem when having 2 or more images in the support set. Nevertheless, it shows that for the best performance, all of the objects should be detected. This can be introduced by labelling some boxes or perhaps by selecting more optimal hyperparameters and prompts for GroundingDINO [5].

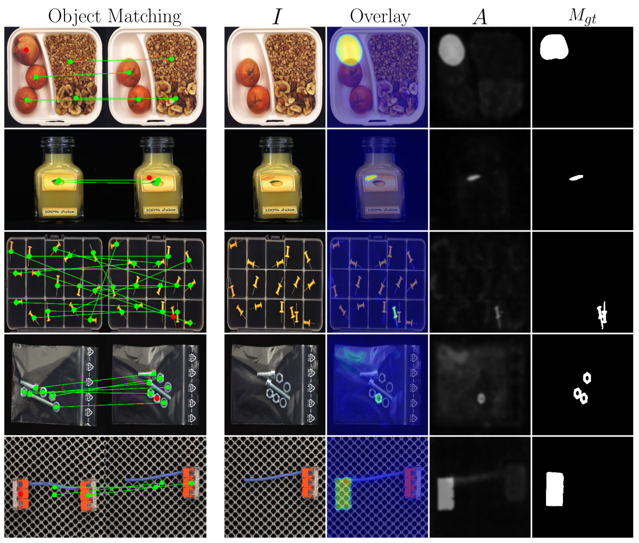

Figure 3. ObjectCore captures composition information required for efficient detection of logical anomalies by creating a representation for every object in the image. During inference, logical anomalies are efficiently detected by matching object representations between a support set image and a query image.

## G. Object Matching visualizations

For a qualitative visualisation of the matches produced through Hungarian matching, we visualised some matches in conjunction with the produced anomaly maps in Figure 3.

## H. Pseudo-detection visualizations

For an even better understanding of the choices we made during the development of the method, we visualized the pseudo-detections made with different parameters concerned with the pseudo-detection extraction.

**Different Thresholds** We have visualized the pseudo-detections created by five different thresholds: 0.15, 0.2, 0.25, 0.3 and 0.35 in Figure 4. When the thresholds are lower, GroundingDINO often detects parts of objects irrelevant to the end task. In contrast, when they are high, many objects are missed. The optimal threshold is usually somewhere between 0.2 and 0.3. The threshold of 0.25 represents a nice middle-ground for general use without any additional knowledge about the objects.

**Different Text Prompts** The pseudo-detections generated by three different prompts (“`item`”, “`object`” and “`part`”) can be seen in Figure 5. All three prompts tend to detect most objects in the image, but the prompt “`part`” can also detect smaller objects that might seem to be a part of something else. This also visually shows that “`part`” is the best choice, but that also “`item`” and “`object`” work fine and could be useful on specific datasets.

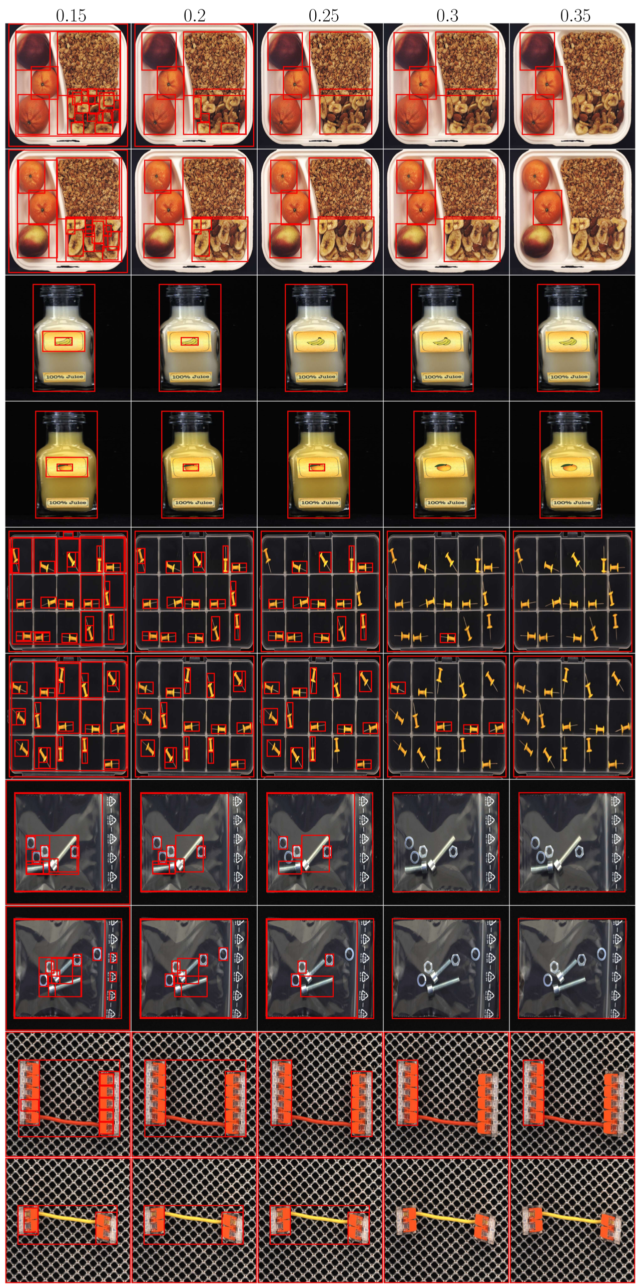

Figure 4. Examples of pseudo-detections produced by five different thresholds. The smaller thresholds also detect smaller parts of objects, introducing noise to the pseudo-detections.

## I. Additional qualitative results

In this section, we add additional qualitative comparisons against state-of-the-art methods Patchcore [6] and PromptAD [4]. The results can be seen in Figures 6, 7, 8 and 9. ObjectCore can detect a larger portion of logical anomalies than previous methods. It also detects structural anomalies as good, if not even better than, previous methods.

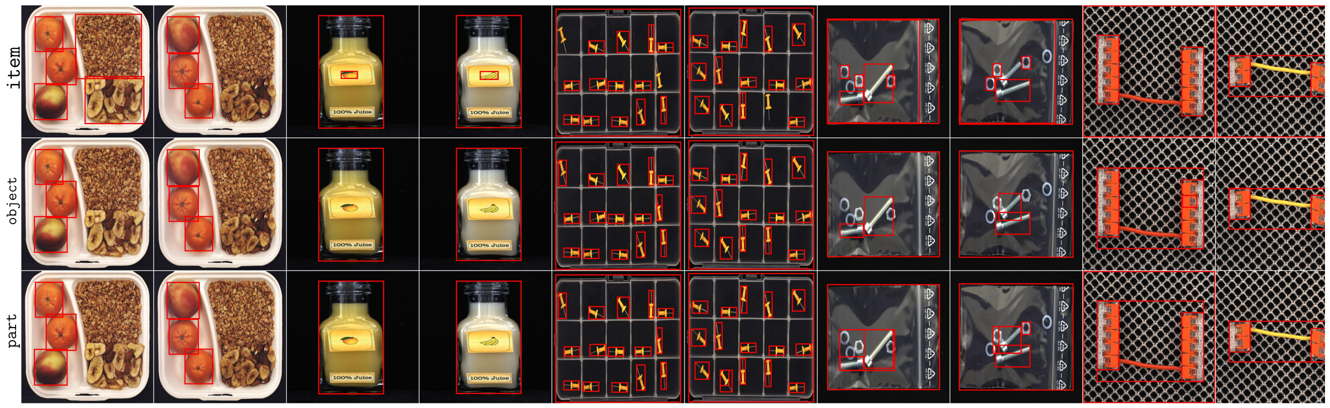

Figure 5. Examples of pseudo-detections produced by three different prompts: "`item`", "`object`" and "`part`". The biggest differences are seen in the breakfast box and screw bag categories.

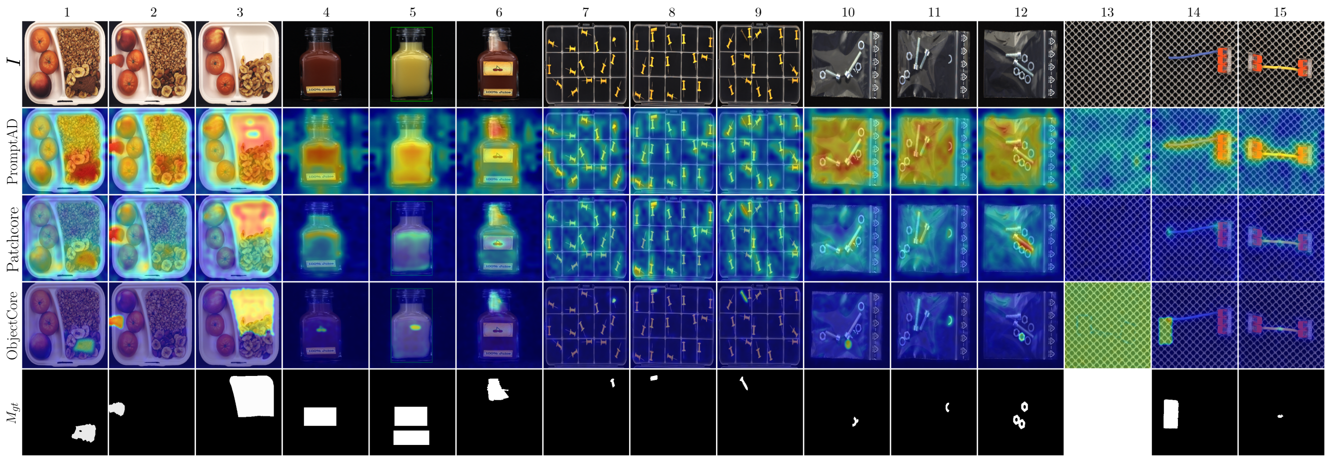

Figure 6. Qualitative comparison of the anomaly segmentation masks on MVTec LOCO [2] produced by ObjectCore and two other state-of-the-art methods. In the first row, the image is shown. In the next three rows, the anomaly segmentations produced by PromptAD [4], Patchcore [6] and ObjectCore are depicted, and in the last row, the ground truth mask is depicted.

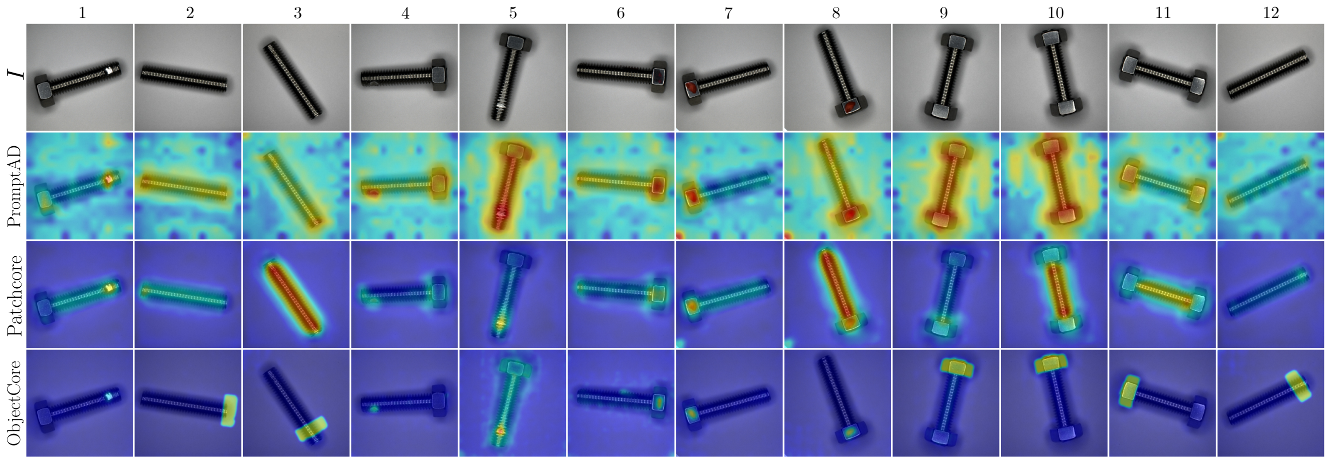

Figure 7. Qualitative comparison of the anomaly segmentation masks on CAD-SD [2] produced by ObjectCore and two other state-of-the-art methods. In the first row, the image is shown. In the next three rows, the anomaly segmentations produced by PromptAD [4], Patchcore [6] and ObjectCore are depicted.

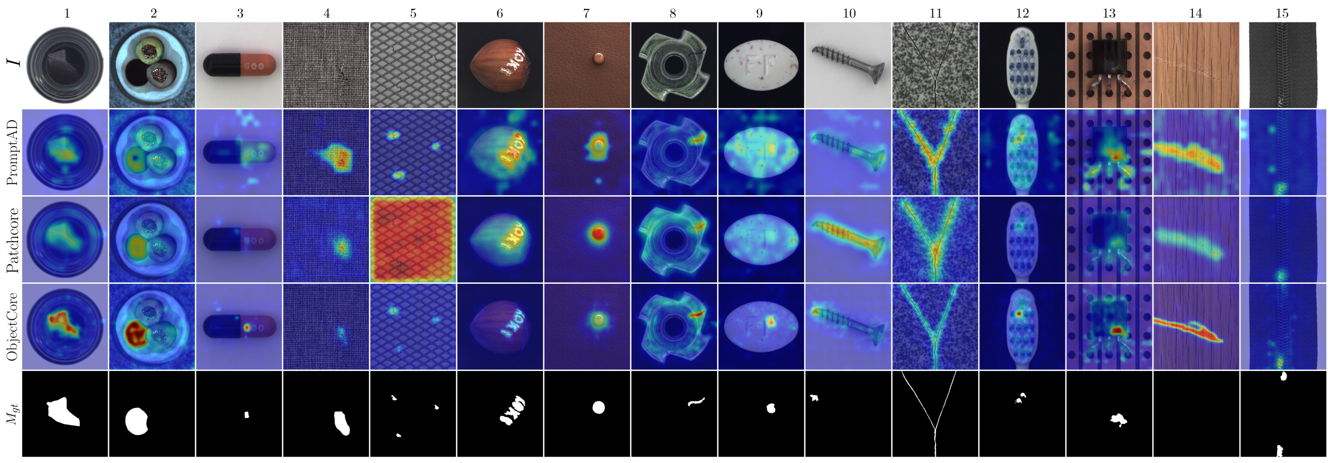

Figure 8. Qualitative comparison of the anomaly segmentation masks on MVTec AD [1] produced by ObjectCore and two other state-of-the-art methods. In the first row, the image is shown. In the next three rows, the anomaly segmentations produced by PromptAD [4], Patchcore [6] and ObjectCore are depicted, and in the last row, the ground truth mask is depicted.

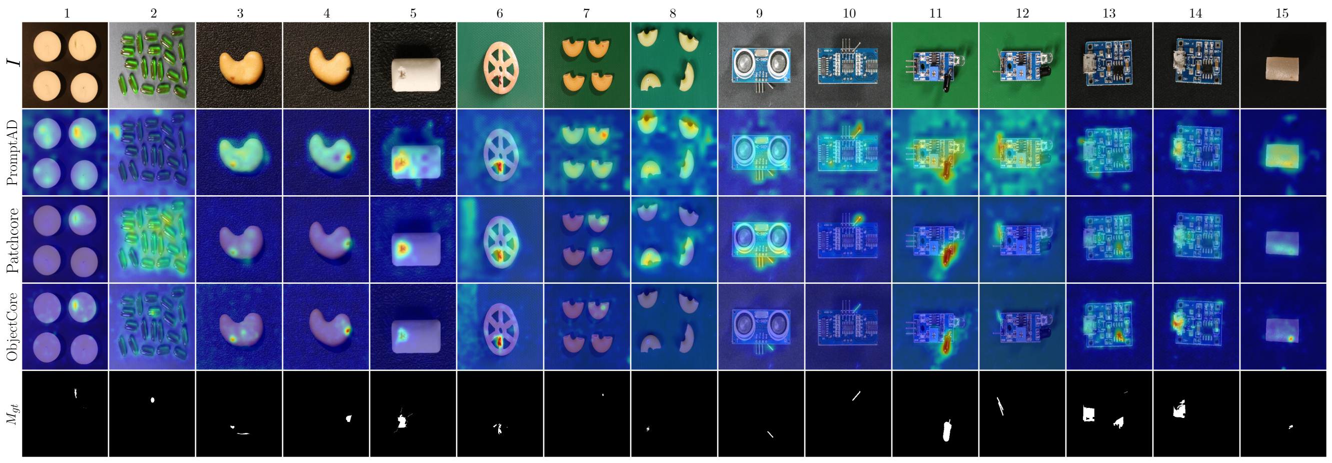

Figure 9. Qualitative comparison of the anomaly segmentation masks on VisA [7] produced by ObjectCore and two other state-of-the-art methods. In the first row, the image is shown. In the next three rows, the anomaly segmentations produced by PromptAD [4], Patchcore [6] and ObjectCore are depicted, and in the last row, the ground truth mask is depicted.

## References

[1] Paul Bergmann, Kilian Batzner, Michael Fauser, David Sattlegger, and Carsten Steger. The MVTec Anomaly Detection Dataset: A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection. *International Journal of Computer Vision*, 129(4):1038–1059, 2021. 6

[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, 5

[3] Kengo Ishida, Yuki Takena, Yoshiki Nota, Rinpei Mochizuki, Itaru Matsumura, and Gosuke Ohashi. Sa-PatchCore: Anomaly Detection in Dataset with Co-Occurrence Relationships Using Self-Attention. *IEEE Access*, 11:3232–3240, 2023. 2

[4] 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*, pages 16838–16848, 2024. 4, 5, 6

[5] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection. In *European Conference on Computer Vision*, pages 38–55. Springer, 2025. 1, 3

[6] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards Total Recall in Industrial Anomaly Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14318–14328, 2022. 4, 5, 6

[7] Yang Zou, Jongheon Jeong, Latha Pemula, Dongqing Zhang, and Onkar Dabeer. SPot-the-Difference Self-supervised Pre-training for Anomaly Detection and Segmentation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXX*, pages 392–408. Springer, 2022. 6