Supplementary Material:
DeSTSeg: Segmentation Guided Denoising Student-Teacher for Anomaly Detection

A. Implementation Details

Image augmentation. All the images are first resized to 256×256. For *bottle*, *grid*, *hazelnut*, *leather*, *tile*, *carpet*, and *screw*, we perform rotation in $\theta \pm 5$ degrees, where θ is randomly chosen from $\{0, 90, 180, 270\}$. For the *wood*, *zipper*, and *cable* categories, we randomly rotate the image in ± 5 degrees, since normal samples of these categories must be aligned to a specific direction, while a rotation of fewer than 5 degrees is allowed. The rotation is not applied to the other categories since their positions, and directions must be identical in all normal images.

Optimization. We use SGD with a momentum of 0.9 and weight decay of 0.0001 as the optimizer. The learning rate of the denoising student network is 0.4, whereas the learning rates of residual blocks and the ASPP segmentation head in the segmentation network are 0.1 and 0.01, respectively. The batch size is 32. The hyperparameter γ in L_{fl} is set to 4, and T for aggregating segmentation results to image-level anomaly score is set to 100. We train the denoising student network for 1,000 iterations and the segmentation network for 4,000 iterations.

Sensitivity of hyperparameter T and γ. In our experiments, the hyperparameters were chosen empirically rather than through tuning. As the MVTec AD training set contains only normal samples, applying a grid search of hyperparameters to estimate the performance with the anomaly data is difficult. Nevertheless, we evaluated several T and γ values, and the results in Tab. S1 and Tab. S2 showed that the performance was consistent across a range of values.

Table S1. Image-level AUC with different values of hyperparameter T.

T	20	50	100(adopted)	200
img (AUC)	98.7	98.6	98.6$\pm$0.4	98.5

Table S2. Image-level AUC, pixel-level AP, and instance-level IAP with different values of hyperparameter γ.

γ	1	2	4(adopted)	8
img (AUC)	98.3	98.0	98.6$\pm$0.4	98.0
pix (AP)	76.2	74.2	75.8$\pm$0.8	75.2
ins (IAP)	75.8	74.3	76.4$\pm$1.0	76.3

Ablation study on data augmentation strategy. In Tab. S3, we examine the effectiveness of the proposed category-specific data augmentation strategy. Results show that the data augmentation strategy can improve performance. In addition, it can be found that even without the data augmentation trick, our model can still achieve the highest pixel-level AP and instance-level IAP among compared methods [1–6], and the image-level AUC is still acceptable. We conclude that the data augmentation strategy could improve the performance of DeSTSeg, while the performance improvement mainly comes from the model design.

B. Training and Inference Time

Our model is built based on the ResNet18 backbone, a relatively small network. Therefore, our model can achieve satisfactory training and inference speed, which is crucial in practice. On a single NVIDIA Tesla V100 GPU, we compare

	img (AUC)	pix (AP)	ins (IAP)
w/o data augmentation	97.7	73.7	72.7
w/ data augmentation	**98.6**	**75.8**	**76.4**

Table S3. Ablation studies on the data augmentation: AUC, AP, and IAP (%) are used to evaluate image-level, pixel-level, and instance-level detection, respectively.

the training time of each category on the MVTec AD dataset and the inference time per image with several methods. Results are shown in Tab. S4.

	training time (min)	inference time (ms)
STPM [4]	15.4	3.1
DRAEM [5]	158.3	22.2
PatchCore [3]*	-	>11.3
Ours	51.2	9.4

* PatchCore does not require to train a network. The reported inference time is only for feature extraction. The total inference time should consider the feature similarity search process, which depends on the memory bank size.

Table S4. Comparison of training and inference time.

C. Detail results of image-level and pixel-level AD

We show the detail results of image-level anomaly detection in Tab. S5.

	US [1]	STPM [4]	CutPaste [2]	DRAEM [5]	DSR [6]	PatchCore [3]	Ours
bottle	99.0	**100.0**	98.3	99.2	**100.0**	**100.0**	**100.0**±0.0
cable	86.2	93.0	80.6	91.8	93.8	**99.6**	97.8±0.5
capsule	86.1	85.9	96.2	98.5	88.4	**99.4**	97.0±0.9
carpet	91.6	99.4	93.1	97.0	**100.0**	98.3	98.9±1.0
grid	81.0	99.8	99.9	99.9	**100.0**	99.1	99.7±0.7
hazelnut	93.1	**100.0**	97.3	**100.0**	95.6	**100.0**	99.9±0.2
leather	88.2	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**±0.0
metal_nut	82.0	**99.9**	99.3	98.7	98.5	**99.9**	99.5±0.5
pill	87.9	89.3	92.4	**98.9**	97.5	96.3	97.2±0.7
screw	54.9	89.4	86.3	93.9	96.2	**97.2**	93.6±2.6
tile	99.1	97.5	93.4	99.6	**100.0**	**100.0**	**100.0**±0.0
toothbrush	95.3	88.1	98.3	**100.0**	99.7	90.3	99.9±0.2
transistor	81.8	95.0	95.5	93.1	97.8	**99.8**	98.5±1.3
wood	97.7	99.1	98.6	99.1	96.3	**98.9**	97.1±1.9
zipper	91.9	90.3	99.4	**100.0**	**100.0**	99.1	**100.0**±0.0
average	87.7	95.1	95.2	98.0	98.2	98.5	**98.6**±0.4

Table S5. Image-level anomaly detection results on MVTec AD dataset (AUC%) with all categories.

D. Instance-level AD under PRO metric

For the instance-level anomaly detection task, we also evaluate the results under PRO [1]. The results are shown in Tab. S6.

	STPM [4]	DRAEM [5]	PatchCore [3]	Ours
bottle	96.4	**97.1**	96.0	96.6±0.6
cable	84.3	75.8	**94.7**	86.4±1.8
capsule	93.9	91.1	**95.9**	94.2±1.1
carpet	**97.1**	93.1	95.5	93.6±3.1
grid	97.0	**98.4**	94.1	96.4±0.7
hazelnut	96.7	**98.7**	96.2	97.6±0.6
leather	97.6	97.9	97.9	**99.0**±0.1
metal_nut	94.2	94.0	**95.7**	95.0±0.7
pill	93.3	88.6	**96.2**	95.3±2.2
screw	93.9	**98.2**	97.3	92.5±1.3
tile	87.3	**98.7**	87.7	95.5±0.9
toothbrush	92.0	90.4	91.0	**94.0**±1.2
transistor	68.1	81.4	**91.0**	85.7±3.1
wood	91.9	93.8	91.3	**96.1**±0.6
zipper	94.5	96.2	96.6	**97.4**±0.7
average	91.9	92.9	**94.5**	94.4±0.4

Table S6. Instance-level anomaly detection results on MVTec AD Dataset (PRO%).

E. Visualization examples of DeSTSeg

For each category in the MVTec AD dataset, we show two examples[1] to illustrate the localization ability of our model in Fig. S1, Fig. S2, and Fig. S3.

[1] All samples shown in this paper are licensed under the CC BY-NC-SA 4.0.

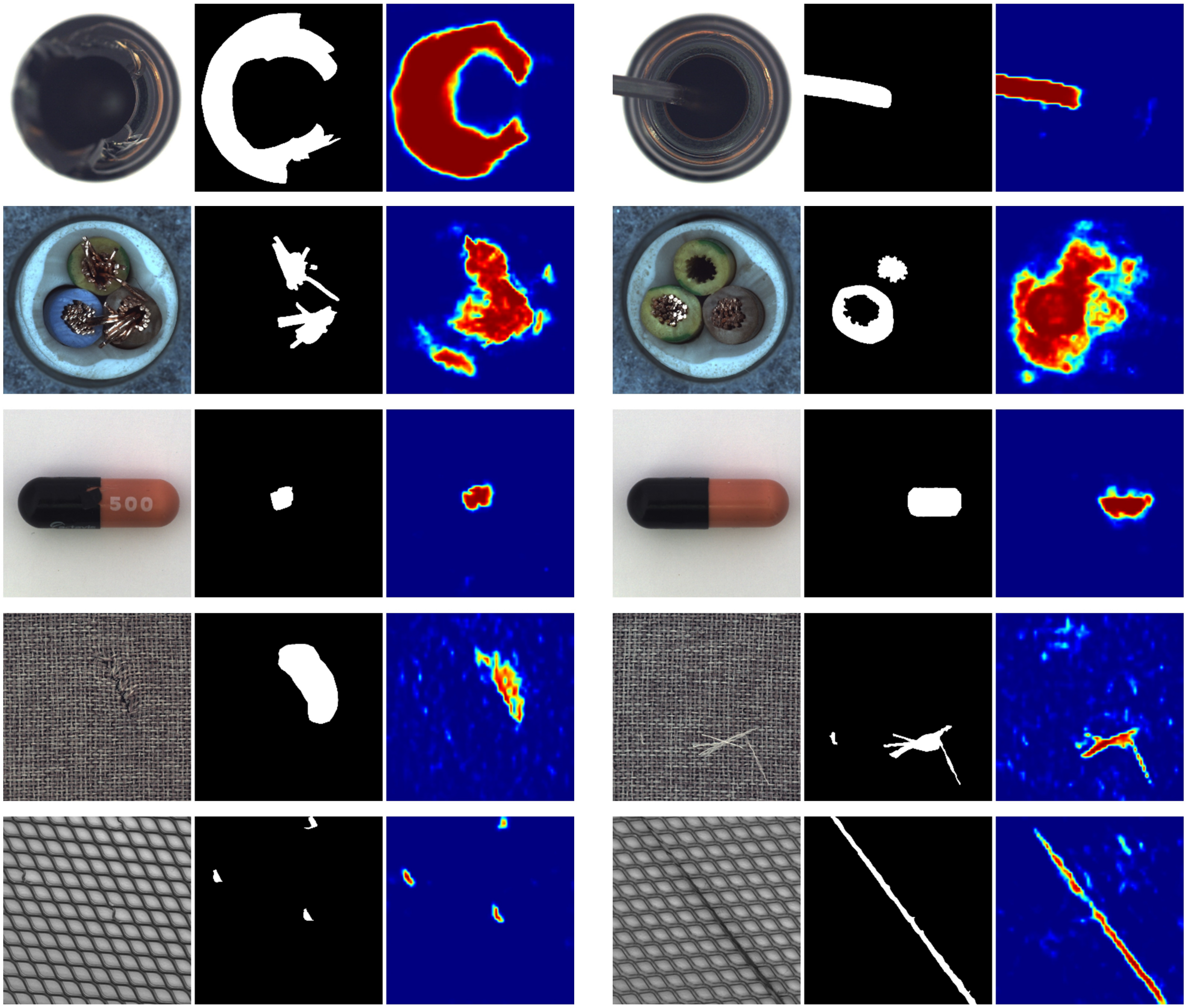

Figure S1. Visualization examples of bottle, cable, capsule, carpet, and grid, from top to bottom.

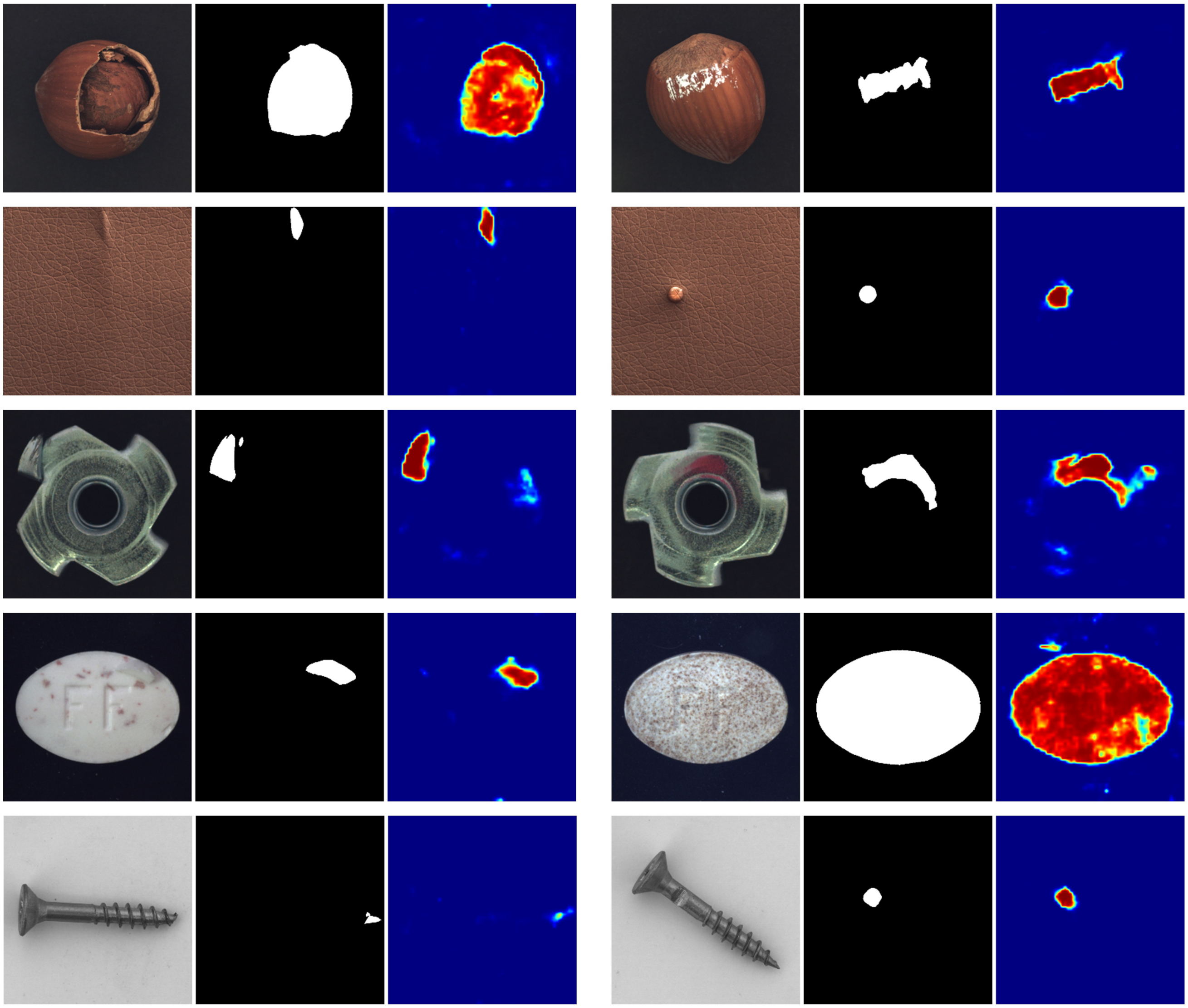

Figure S2. Visualization examples of hazelnut, leather, metal_nut, pill, and screw, from top to bottom.

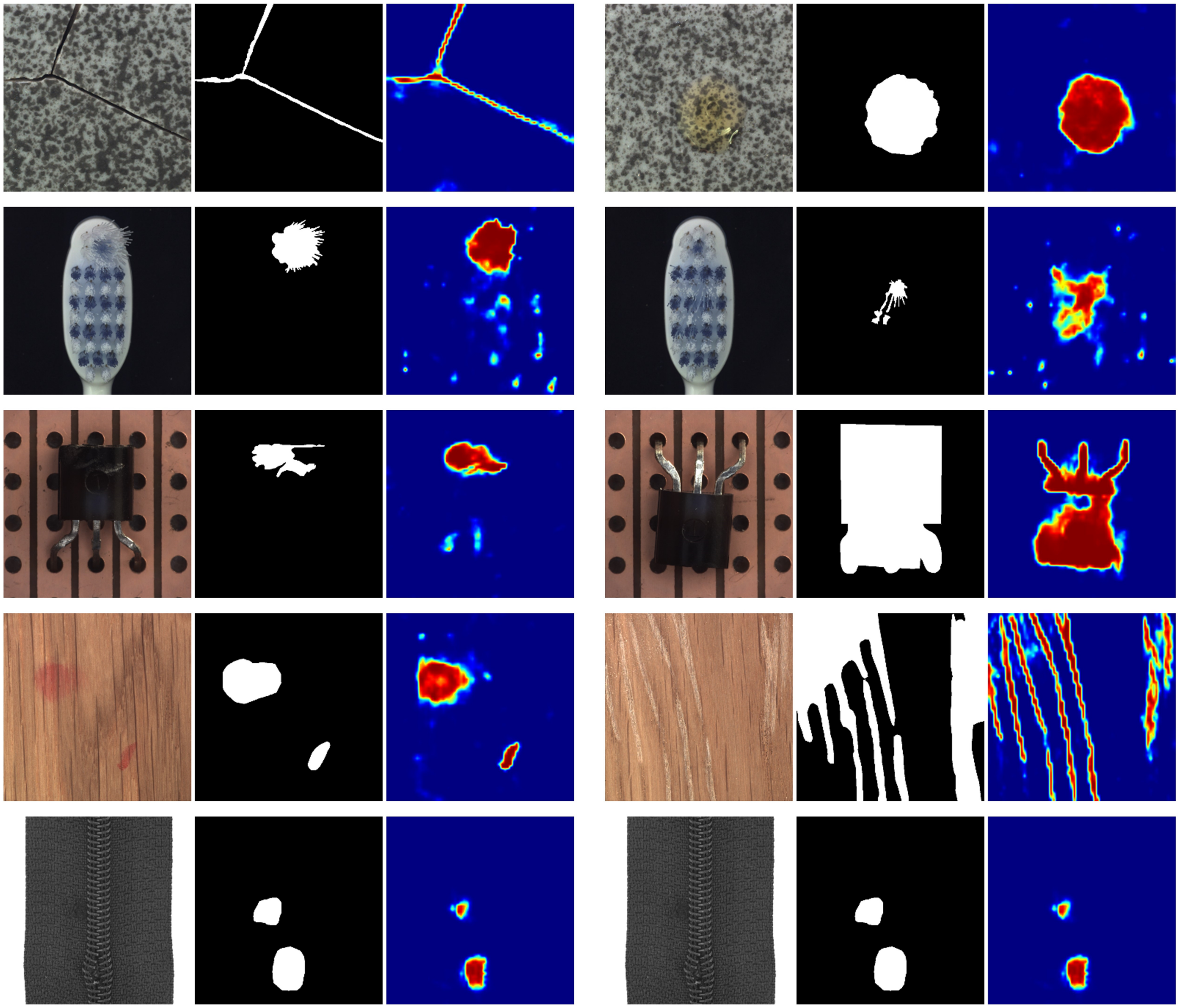

Figure S3. Visualization examples of tile, toothbrush, transistor, wood, and zipper, from top to bottom.

References

[1] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4183–4192, 2020. 1, 2

[2] Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9664–9674, 2021. 1, 2

[3] 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. 1, 2, 3

[4] Guodong Wang, Shumin Han, Errui Ding, and Di Huang. Student-teacher feature pyramid matching for anomaly detection. In *Proceedings of the British Machine Vision Conference*, 2021. 1, 2, 3

[5] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. Draem-a discriminatively trained reconstruction embedding for surface anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8330–8339, 2021. 1, 2, 3

[6] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. Dsr–a dual subspace re-projection network for surface anomaly detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 539–554. Springer, 2022. 1, 2