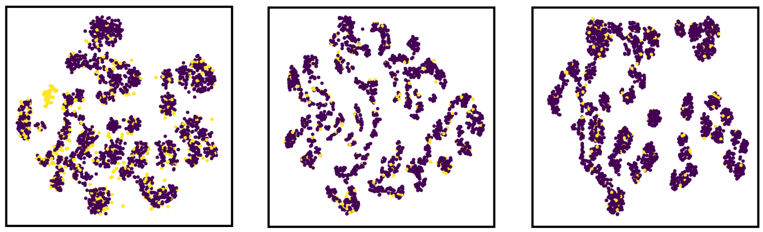
A Appendix

A.1 Visualization of Coreset

Figure 5 shows the visualization of coreset in the memory bank. The coreset of SoftCore is clean and decentralized.



(a) PatchCore (b) PatchCore-Random (c) SoftCore-LOF

Figure 5: Comparison between corsets of AD methods with same noisy train set, MVTecAD-Pill with noise-0.1. We use t-SNE for dimension reduction for visualization. The yellow dots represent patch features from noisy sample, while the purple dots are nominal. Compared with the other two, SoftCore wipe off the noisy patch and model the nominal data properly.

A.2 Details of Experimental Results

Table 5: Anomaly localization performance details of all classes. The results are evaluated on MVTecAD-noise-0.1.

Noise=0.1	No overlap						Overlap				
Category	PaDiM	CFLOW	PatchCore	SoftCore-nearest	SoftCore-gaussian	SoftCore-lof	PaDiM*	CFLOW	PatchCore	PatchCore 1%-Random	SoftCore-lof
bottle	0.986	0.984	0.987	0.987	0.986	0.987	0.981	0.984	0.714	0.979	0.975
cable	0.916	0.958	0.843	0.915	0.981	0.983	0.946	0.950	0.670	0.969	0.971
capsule	0.986	0.985	0.986	0.988	0.977	0.990	0.984	0.986	0.883	0.984	0.989
carpet	0.992	0.989	0.992	0.992	0.993	0.992	0.980	0.986	0.765	0.951	0.989
grid	0.974	0.947	0.991	0.990	0.989	0.990	0.879	0.961	0.482	0.882	0.974
hazelnut	0.987	0.991	0.990	0.990	0.991	0.990	0.978	0.982	0.418	0.957	0.924
leather	0.994	0.994	0.991	0.994	0.994	0.993	0.992	0.993	0.683	0.987	0.993
metal_nut	0.933	0.956	0.842	0.894	0.964	0.984	0.911	0.960	0.779	0.938	0.983
pill	0.956	0.983	0.971	0.974	0.972	0.981	0.960	0.976	0.608	0.971	0.976
screw	0.989	0.977	0.995	0.991	0.969	0.994	0.974	0.973	0.745	0.953	0.969
tile	0.956	0.953	0.953	0.960	0.962	0.954	0.921	0.945	0.700	0.919	0.954
toothbrush	0.991	0.988	0.989	0.988	0.988	0.985	0.954	0.984	0.692	0.984	0.985
transistor	0.960	0.887	0.847	0.965	0.954	0.942	0.939	0.834	0.317	0.914	0.936
wood	0.973	0.964	0.969	0.947	0.946	0.939	0.946	0.934	0.522	0.896	0.929
zipper	0.986	0.978	0.986	0.989	0.988	0.988	0.978	0.979	0.823	0.975	0.986

A.3 Performance Trends in Noise

Figure 6 and Figure 7 show the performance trends of SOTA AD methods and softcore in different noisy scenes. Since overconfident in the training data and the greedy subsampling algorithm, PatchCore performance decreases most obviously with the noise increase. In contrast, CFLOW and PaDiM are also affected by noise, but the amplitudes are smaller. SoftCore maintains a consistent level of performance at all noise levels. Unfortunately, SoftCore is slightly weaker than PatchCore in noiseless scenes, which may be due to the excessively conservative threshold setting.

A.4 Image-level Denoising V.S. Patch-level Denoising

A simple strategy to eliminate the noisy data is to delete the anomaly samples before training. This is an unsupervised outlier detection task. However, the existing outlier detection methods do not work well because the distance between abnormal and normal images is much smaller than the distance

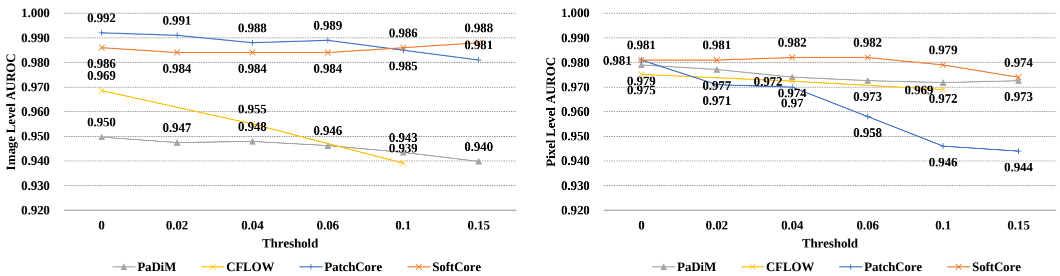


Figure 6: Performance in different level no overlap noise.

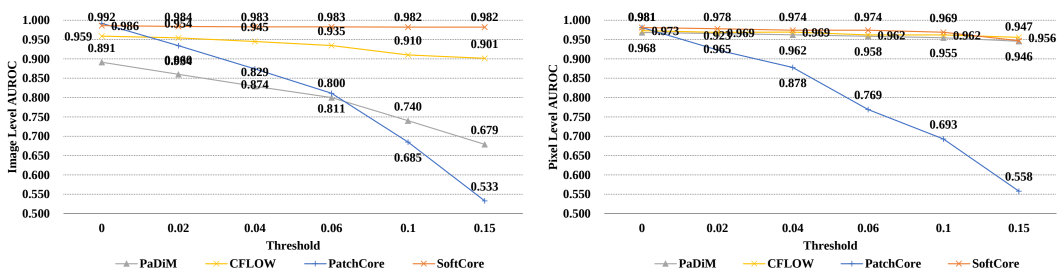


Figure 7: Performance in different level overlap noise.

between different classes. Meanwhile, we found that some AD methods could also detect outliers in the training set. After weighing the costs and effects, we apply PaDiM* as the image-level denoising method to compare with patch-level denoising Softcore. PaDiM* is a simplified version of PaDiM, which uses ResNet18 as the backbone with faster computing speed. PaDiM* scores all training samples and then removes the pieces with high outliers based on the threshold. The comparison in Table 6 and Table 7 show that image-level denoising dramatically improves the performance of existing SOTA AD methods in the noisy scene. But there is still a gap when compared with SoftCore.

Table 6: The anomaly detection performance of image-level denoising and patch-level denoising. The PaDiM*+PaDiM*, PaDiM*+CFLOW, and PaDiM*+PatchCore are AD methods with image-level denoising. PaDiM* is used in image level denoising, where we use the same threshold (0.15) as it in softcore. And we also tried the tricky threshold-0.1 as the noise ratio, but it works worse. The results are evaluated on MVTecAD-noise-0.1 with overlap.

Category	PaDiM*	PaDiM*+ PaDiM*	CFLOW	PaDiM*+ CFLOW	PatchCore	PaDiM*(threshold -0.1)+PatchCore	PaDiM*+ PatchCore	SoftCore-lof
bottle	0.937	0.994	**1.000**	**1.000**	0.692	0.984	**1.000**	**1.000**
cable	0.680	0.741	0.916	0.841	0.756	0.890	0.888	**0.994**
capsule	0.796	0.854	0.945	0.939	0.783	0.892	0.909	**0.955**
carpet	0.890	0.937	0.960	0.950	0.681	0.963	0.974	**0.993**
grid	0.674	0.765	0.799	0.830	0.526	0.850	0.870	**0.969**
hazelnut	0.543	0.725	0.999	0.990	0.441	0.871	0.929	**1.000**
leather	0.964	0.979	0.996	**1.000**	0.739	0.957	0.989	**1.000**
metal_nut	0.820	0.949	0.957	0.986	0.765	0.965	0.977	**1.000**
pill	0.722	0.745	0.897	0.924	0.770	0.898	0.913	**0.955**
screw	0.567	0.542	0.570	0.639	0.710	0.916	0.907	**0.923**
tile	0.830	0.906	0.980	0.981	0.716	0.939	0.957	**0.981**
toothbrush	0.700	0.869	0.878	0.928	0.800	0.981	**0.997**	0.994
transistor	0.471	0.770	0.872	0.788	0.491	0.777	0.825	**0.999**
wood	0.831	0.966	0.954	0.970	0.579	0.943	0.976	**0.986**
zipper	0.679	0.678	0.931	0.873	0.792	0.909	0.914	**0.974**
Average	0.740	0.828	0.910	0.909	0.683	0.916	0.935	**0.982**

Table 7: The anomaly localization performance of image-level denoising and patch-level denoising.

Category	PaDiM*	PaDiM*+ PaDiM*	CFLOW	PaDiM*+ CFLOW	PatchCore	PaDiM*(threshold -0.1)+PatchCore	PaDiM*+ PatchCore	SoftCore-lof
bottle	0.937	0.983	**1.000**	0.986	0.692	0.984	0.985	**1.000**
cable	0.680	0.954	0.916	0.956	0.756	0.738	0.739	**0.994**
capsule	0.796	0.982	0.945	**0.985**	0.783	0.851	0.876	0.955
carpet	0.890	0.984	0.960	0.988	0.681	0.960	0.988	**0.993**
grid	0.674	0.876	0.799	0.948	0.526	0.797	0.818	**0.969**
hazelnut	0.543	0.977	0.999	0.987	0.441	0.798	0.825	**1.000**
leather	0.964	0.993	0.996	0.995	0.739	0.966	0.979	**1.000**
metal_nut	0.820	0.968	0.957	0.984	0.765	0.784	0.834	**1.000**
pill	0.722	0.956	0.897	**0.984**	0.770	0.706	0.713	0.955
screw	0.567	0.968	0.570	**0.970**	0.710	0.887	0.889	0.923
tile	0.830	0.927	0.980	0.946	0.716	0.924	0.968	**0.981**
toothbrush	0.700	0.986	0.878	0.983	0.800	0.977	0.986	**0.994**
transistor	0.471	0.965	0.872	0.908	0.491	0.932	0.945	**0.999**
wood	0.831	0.947	0.954	0.943	0.579	0.800	0.918	**0.986**
zipper	0.679	0.973	0.931	0.967	0.792	0.875	0.878	**0.974**
Average	0.740	0.963	0.910	0.969	0.683	0.865	0.889	**0.982**

Table 8: Mean training and inference time per category on MVTecAD. The unit of time is second.

	Training time	Inference time
SoftCore-LOF	21.2958	15.6146
PatchCore	21.3869	15.8763
PaDiM*+PatchCore	74.2912	15.5386

A.5 Training and Inference Time

Excluding the loading time of data, the comparison of the remaining time overhead between SoftCore and PatchCore is shown in Figure 8. The GPU used in this experiment is RTX TITAN 24G. Both spend almost the same amount of time training and testing, which means that our patch-level denoising does not bring unacceptable overhead. On the contrary, the image-level denoising dramatically increases training time.

A.6 The Noise in Existing Dataset

For example, MVTec Anomaly Detection benchmark (MVTecAD) dataset [5] is a comprehensive and widely used benchmark for unsupervised abnormal detection. While in such a well-established academic dataset, noisy data also exists in the training set, as in Fig. 8.

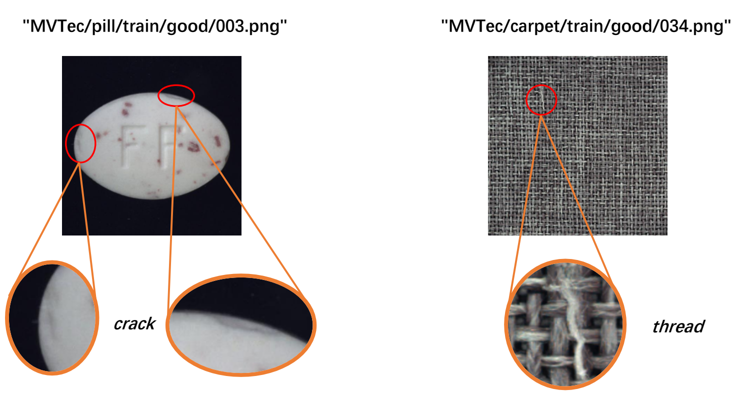


Figure 8: Noisy example in the MVTecAD dataset which is a widely used benchmark for AD research.