Supplementary Document for FUN-AD: Fully Unsupervised Learning for Anomaly Detection with Noisy Training Data

Jiin Im[1*] Yongho Son[2*] Je Hyeong Hong[1,2†]
[1]Dept. Electronic Engineering, Hanyang University [2]Dept. Artificial Intelligence, Hanyang University

1. Additional statistical analysis of pairwise distances between features

Empirical validation Since our statistical analysis is limited to isotropic Gaussian distributions, it is not directly applicable other distributions or real-world data. Therefore, we aim to bridge this theoretical gap with empirical analysis using real-world data. We validate these findings on both synthetic data with isotropic Gaussian distributions and real data from the *bottle* set in MVTec AD [1], utilizing normal images from the training set and anomaly images from the test set, as the training set does not contain any anomalies.

For the synthetic experiment, we sampled 1000 16-dimensional features from the normal distribution $\mathcal{N}(\mathbf{0}, \mathtt{I})$ and 1000 samples from the anomaly distribution $\mathcal{N}(\mathbf{1}.5, 2\mathtt{I})$. We then computed all pairwise feature distances, resulting in the histogram shown in Fig. 1a. For the real experiment, we extracted both image-level features and patch-level features from all 209 normal (training) images and 63 anomaly (test) images of the *bottle* sequence in MVTec AD using the pretrained DINO model from [2]. Again, we calculated all pairwise feature distances over all pairs of patch-level features and over all pairs of image-level features, yielding histograms in Figs. 1b and 1c. While the histograms have different degrees of skewness, we observe the normal pairs are consistently the most likely to yield shorter pairwise distances compared to other types of pairs.

2. Toy example of semantic anomaly detection

We used CIFAR-10 [7] to conduct a toy experiment, setting the data to a scenario where the distribution of outliers is more spread out than the distribution of normals, consistent with our assumptions. The normal class is "automobile", and the outliers consist of the remaining classes in CIFAR-10. The contamination ratio (the ratio of outliers to normals) within the training dataset is set to 10%. Unlike detecting patch-level defects, Local-Net in semantic anomaly detection outputs one anomaly score per image (because semantic anomalies are not divided into normal and abnormal regions within a single image). For further details, please refer to Sec. 4 for related results.

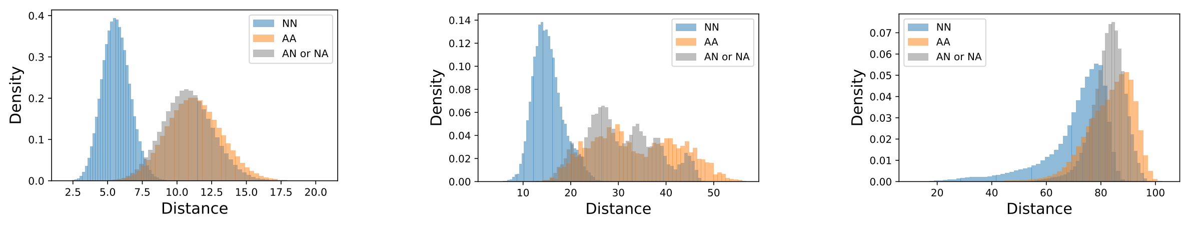
3. Additional framework details

Overall architecture FUN-AD comprises two subnetworks: a pretrained feature extractor $\mathcal{E}$ (to leverage semantic information, the self-supervised DINO [2]) based on vision transformer (ViT) [4] and the Local-Net model ϕ based on a simple multilayer perceptron (MLP) for detecting patch-level anomalies. $\mathcal{E}$ takes an image I_i as input and outputs one class token and P patch tokens. To identify anomalies at the patch level, we concatenate the class token with each patch token to form a patch-level feature $\mathbf{f}_{ij} \in \mathbb{R}^D$ for each image patch X_{ij}. With abuse of notation, we represent $I_i = \{X_{ij}\}_{j=1}^P$, where $P = HW/K^2$. In our setting, $H = W = 224$, $K = 8$, and thus $P = 784$. Since the class token is 768-dimensional and the patch token is 768-dimensional, $D = 1536$. The local patch feature $\mathbf{f}_{ij}$ serves as input to the Local-Net ϕ, from which we obtain a normalized anomaly score using a sigmoid function.

Model inference In the inference phase, FUN-AD performs anomaly detection and anomaly localization by predicting the anomaly score for a given input. For anomaly detection, a test image X_i is passed through $\mathcal{E}$ to extract the patch-level features $\{\mathbf{f}_{ij}\}_{j=1}^P$. These features are passed through ϕ to obtain the patch-level anomaly scores. We then perform global max-pooling of these scores to calculate the image-level global anomaly score. For anomaly localization, the patch-level anomaly scores are spatially arranged to form an anomaly score map, as shown in Fig. 5 in [6]. As in [8,15], we then perform bilinear interpolation of the map with Gaussian smoothing ($\sigma = 4$) to match the dimensions of the original image ($H \times W$).

Implementation details The Local-Net has the FC[1536, 1024, 128, 1] structure. Leaky ReLU activation functions (slope: 0.2) are applied between layers, and the output layer uses the sigmoid function for outputting normalized anomaly score. We use the RMSProp optimizer with momentum of 0.2 and learning rate of 2e-5 for training using the batch size of 32. We set $\lambda = 2.5$, $\tau_b = 0.5$, $\tau_n = 0.5$

*indicates equal contributions.
†Corresponding author



(a) Synthetic (isotropic Gaussians) (b) Real (bottle, image features) (c) Real (bottle, patch features)

Figure 1. Histogram of pairwise distances for different types of feature pairs. Abbreviations are as follows: NN for normal-normal pairs, AA for anomaly-anomaly pairs, AN or NA for anomaly-normal or normal-anomaly pairs. For the synthetic experiment, the normal samples were drawn from $\mathcal{N}(\mathbf{0}, \mathtt{I})$ and the anomaly samples from $\mathcal{N}(\mathbf{1.5}, 2\mathtt{I})$. For the "real" comparison, we used patch features (patch tokens) and image features (class tokens) extracted from the pretrained DINO model [2] for the bottle set [1].

Dataset	Method	Anomaly-to-normal ratio					
		0%	1%	3%	5%	10%	20%
MTD [5]	InReach [9]	83.55 / 72.02	84.08 / 75.49	80.30 / 78.79	80.93 / 78.64	80.73 / 72.23	88.42 / 81.91
	SoftPatch [15]	76.11 / 90.63	77.19 / 91.53	79.61 / 92.67	80.51 / 91.48	83.61 / 87.52	85.26 / 93.49
	FUN-AD	79.55 / 94.75	82.51 / **94.58**	85.87 / 94.97	85.35 / 93.84	85.12 / 93.52	94.74 / 95.37
	FUN-AD*	**83.61 / 93.51**	**85.25** / 93.59	**87.52 / 95.22**	**85.46 / 94.13**	**85.28 / 93.59**	**95.79 / 97.76**

Table 1. Performance comparison of different fully-unsupervised anomaly detection methods across different anomaly-to-normal ratios on the contaminated MTD dataset (*no overlap*). * indicates synthetic anomaly data has been utilized for training. The best results are in bold and the runner-ups are underlined.

and $\tau_c = 0.9$ by default. For each object/texture class, we train for 1500 epochs and choose the model with the best average of image-wise and pixel-wise AUROCs. In Sec. 3 of the main paper [6], the real data consists of normal data in the training set and anomalous data in the test set. Also, we set the patch size to 8, yielding one 768-dimensional image-level feature and 28^2 768-dimensional patch-level features for each normal or anomaly image.

4. Additional ablation studies

Effect of different contamination rates Tab. 1 demonstrates the performance of FUN-AD according to the contamination ratio in the training dataset. Here, "FUN-AD" refers to the results obtained from training with the dataset without synthetic anomalies, while "FUN-AD*" refers to the results from training the FUN-AD framework with synthetic anomalies added at a rate of 5% of the training dataset size. Synthetic anomalies were created from a noisy (anomaly-present) dataset considering a fully unsupervised setting. The results demonstrate that our proposed framework achieves state-of-the-art performance on texture-based dataset, highlighting its robustness across various types of anomalies.

Inference time In an industrial setting, real-time anomaly detection is crucial. When comparing the inference speed with existing methods using the GPU RTX-4090 (refer to Tab. 2), our method operates at an impressive speed of approximately 113 fps, outperforming other methods.

Effect of weight on mutual smoothness loss Tab. 3 shows that optimal performance is achieved when $\lambda = 2.5$ on MVTec AD. In these results, $\lambda = 0$ indicates that the pseudo-labeling method alone is sufficient for the network to learn from the normal and anomaly information and succeed in anomaly detection and localization. Additionally, when mutual-smoothness loss is applied, the anomaly detection performance improves with a weight value of $\lambda = 2.5$ compared to using only pseudo-labeling.

Effects of random sampling rate Since Eq.7 needs to be calculated for pseudo-labeling, the training time overhead can be significant if computed with all feature vectors in the memory bank. However, applying corset sampling [12], which has been used in a one-class classification environment, is difficult because we cannot assume that all the samples in the memory bank are normal. Therefore, we compare the performance by randomly sampling only a small percentage of the feature vectors in the memory bank. Table 4 shows the performance of anomaly detection and localization according to the sampling ratio. The performance does not vary significantly depending on the degree of sampling. This indicates that using a low sampling rate for efficient training does not result in significant performance degradation.

Effects of synthetic supervised loss The comparison with and without y_{syn} in Tab. 5 shows that our proposed pseudo labels perform better than those using Perlin masks to assign labels for synthetic anomalies. This indicates that our pseudo-labeling method is more effective for detecting real anomalies by identifying semantically anomalous regions and using them for training, rather than merely learning that regions with the Perlin noise are anomalous.

Method	FUN-AD (*Ours*)	SoftPatch [15]	InReaCh [9]
Throughput (fps)	**112.7**	22.5	8.8

Table 2. Inference speeds achieved by different fully unsupervised anomaly detection algorithms on the VisA dataset.

λ	AUROC$_{image}$(%)	AUROC$_{pixel}$(%)
0	98.72	98.39
0.25	98.54	98.41
1	98.83	98.45
2.5	**98.95**	**98.55**
10	98.55	98.34

Table 3. Ablation study of weights for the mutual smoothness loss. The optimal performance is achieved when $\lambda = 2.5$ on MVTec AD.

Sampling ratio	AUROC$_{image}$(%)	AUROC$_{pixel}$(%)
0.25	98.83	**98.66**
0.5	98.95	98.55
0.75	**99.11**	98.51
1.0	98.84	98.53

Table 4. Ablation study of the sampling rate when storing normally labeled feature vectors in memory banks. This demonstrates the capability of efficient training with a low sampling rate.

Method	AUROC$_{image}$(%)	AUROC$_{pixel}$(%)
w/o y_{syn}	97.83	**97.51**
w y_{syn}	98.95	98.55

Table 5. Ablation study of synthetic supervised loss. y_{syn} indicates whether the mask of the synthetic anomaly is used or not.

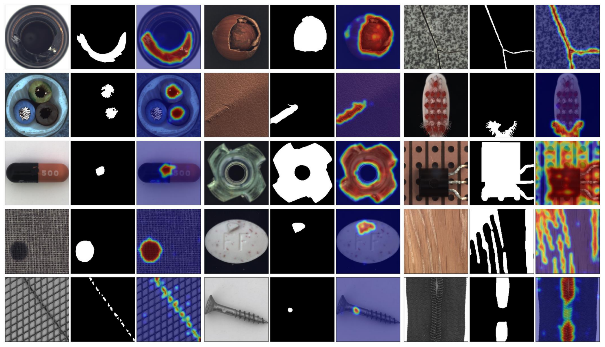
Semantic anomaly detection Detecting semantic anomalies also requires a fully unsupervised setting, and according to [14], it is more similar to the real world when the training data is contaminated with abnormal samples. Therefore, we conducted experiments on the STR-10 [13] and CIFAR-10 [7] datasets to verify the applicability of our framework. We designated one class as the normal class and randomly sampled anomalies from the remaining classes to create contaminated unlabeled datasets with a 1:10 anomaly-to-normal ratio. The findings are presented in Tab. 6, demonstrating that although FUN-AD was originally developed for industrial anomaly detection, it effectively distinguishes between normal and abnormal classes under specific conditions. In these experiments, where the normal class is singular and the abnormal classes encompass the remaining nine, the variation is substantial enough to validate the effectiveness of our assumptions and approach.

5. Qualitative results

Fig. 2 shows some anomaly localization results yielded by FUN-AD. Each class is represented by three columns: the first column shows the RGB image, the second column shows the segmentation mask of the defect area, and the third column shows the anomaly score predicted by FUN-AD. Our method is not only effective at detecting large defects but also excels at clearly separating the boundary between normal and anomaly without ambiguity, even in the presence of very small defects. This is evident when comparing the ground-truth mask and the heatmap in Fig. 2. These results demonstrate that FUN-AD is robust, particularly for very small defects, but not limited to large detects with higher confidence score compared to other models.

Dataset	AUROC (%)
CIFAR-10 [7]	95.10
STL-10 [13]	99.63

Table 6. Average semantic anomaly detection results for scenarios (10, e.g., cat is normal) where each semantic class is normal.


(a) MVTec AD

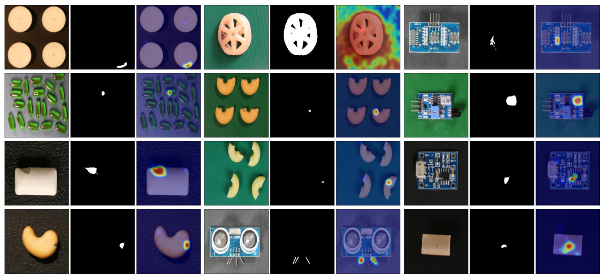

(b) VisA

Figure 2. Visualization of anomaly detection results achieved by FUN-AD on the MVTec AD and VisA datasets. Each binary mask shows the anomaly segmentation map while each heatmap visualizes the anomaly region (red means likely to be an anomaly while blue means unlikely).

6. Details of the experimental results

We show the experimental results for all categories of *overlap*, *No overlap* for MVTec AD and VisA in Tab. 7, 8, 9, 10. Each table presents image-wise AUROC (%) / pixel-wise AUROC (%), representing anomaly detection and localization performance, respectively. The best results are in bold and the runner-ups are underlined. *FUN-*

AD places a stronger emphasis on local anomalies by utilizing Local-Net for inference. Consequently, it excels at detecting small defects in images with multiple instances, as observed in capsules and macaroni2 in VisA, outperforming other models in this regard.

7. Limitations and broader impacts

Limitations While FUN-AD is shown to work across many different unsupervised settings, it may be compromised if the feature diversity of the normal data is comparable to that of the anomalies, e.g. when one type of anomaly dominates. Also, our analytic analysis in Sec.3.1 is limited to the case of normal and anomaly distributions following isotropic Gaussians. Our approach still requires use of a pretrained feature extraction network such as DINO [2] for basic initialization. Finally, FUN-AD yields suboptimal performance for scarce anomaly-to-normal ratios (0 to 1%).

Broader impacts Our approach can reduce the physical burden of human workers by reducing the manual labor required for annotating normal samples. This allows reducing expenditure on data acquisition which in return may be invested towards improving the quality of product.

Type	One-class classification					Fully unsupervised		
Method	CS-Flow [11]	PaDiM [3]	PatchCore [10]	SimpleNet [8]	RealNet [16]	SoftPatch [15]	InReaCh [9]	FUN-AD (*Ours*)
Bottle	98.7 / -	99.3 / 98.4	**100.0** / 98.2	**100.0** / 97.2	**100.0** / 98.2	**100.0** / 98.7	**100.0** / 98.3	**100.0 / 99.2**
Cable	95.5 / -	89.3 / 93.6	96.9 / 80.9	97.0 / 93.7	96.1 / 95.6	**99.1 / 98.7**	94.2 / 97.5	98.7 / 94.0
Capsule	95.0 / -	90.5 / 98.5	97.2 / 98.7	95.3 / 98.1	**99.8** / 98.6	95.8 / **98.9**	49.7 / 93.4	99.7 / 98.2
Carpet	99.8 / -	**100.0** / 99.2	98.7 / 99.1	99.4 / 98.8	98.6 / 97.7	99.1 / 99.4	98.4 / 99.5	**100.0 / 99.7**
Grid	96.5 / -	93.4 / 95.6	96.2 / 98.9	99.1 / 97.9	**100.0 / 99.6**	96.3 / 98.7	91.2 / 97.7	**100.0** / 99.4
Hazelnut	95.1 / -	92.7 / 98.0	**100.0** / 98.8	97.6 / 95.6	99.9 / 98.5	**100.0** / 98.8	98.3 / 97.8	**100.0 / 99.7**
Leather	98.6 / -	**100.0** / 99.3	**100.0** / 99.2	**100.0** / 98.9	**100.0** / 99.7	**100.0** / 99.4	**100.0** / 99.3	**100.0 / 99.8**
Metal_nut	91.8 / -	97.7 / 89.3	99.1 / 77.6	99.0 / 85.8	**100.0** / 87.0	**100.0** / 86.8	96.8 / 95.1	**100.0 / 99.5**
Pill	89.3 / -	93.7 / 96.2	97.0 / 97.0	95.6 / 97.9	**99.1 / 98.9**	96.7 / 97.8	88.6 / 96.0	98.5 / 98.2
Screw	79.3 / -	84.5 / 98.4	95.0 / 98.6	89.0 / 97.8	**98.8 / 99.4**	94.5 / **99.4**	79.7 / 98.3	94.8 / 98.1
Tile	96.2 / -	97.8 / 94.9	99.2 / 92.8	99.4 / 91.6	**99.7** / 97.8	98.7 / 96.3	99.0 / 97.3	99.5 / **98.7**
Toothbrush	92.7 / -	**100.0** / 98.8	99.7 / 98.8	**100.0** / 98.4	98.3 / 95.0	99.7 / 98.6	99.0 / **98.8**	97.9 / 98.7
Transistor	92.3 / -	94.6 / 96.8	96.8 / 87.7	96.0 / 89.7	98.0 / 90.9	**99.6** / 93.6	99.2 / 97.1	99.3 / **97.7**
Wood	90.5 / -	98.1 / 94.3	96.6 / 95.6	99.7 / 92.9	99.9 / 97.3	98.1 / 95.1	95.3 / 92.3	**100.0 / 98.0**
Zipper	95.1 / -	88.2 / 98.4	98.7 / 98.1	98.7 / 95.2	99.6 / 98.9	97.5 / 98.9	93.2 / 94.9	**100.0 / 99.3**
Average	93.8 / -	94.7 / 96.6	98.1 / 94.7	97.7 / 95.3	**99.2** / 96.9	98.3 / 97.3	92.2 / 96.9	**99.2 / 98.6**

Table 7. Detailed results for MVTec AD in the *No overlap* setting.

Type	One-class classification					Fully unsupervised		
Method	CS-Flow [11]	PaDiM [3]	PatchCore [10]	SimpleNet [8]	RealNet [16]	SoftPatch [15]	InReaCh [9]	FUN-AD (*Ours*)
Bottle	67.4 / -	79.9 / 96.0	89.4 / 68.2	76.2 / 46.3	99.3 / 97.7	**100.0** / 93.1	**100.0** / 98.4	**100.0 / 99.2**
Cable	72.7 / -	68.0 / 86.0	87.1 / 65.0	41.2 / 51.7	88.3 / 93.5	**99.3 / 98.4**	94.6 / 97.8	96.9 / 95.4
Capsule	77.7 / -	81.4 / 98.3	88.8 / 92.3	43.0 / 61.0	**97.8 / 99.2**	95.3 / 98.7	52.1 / 94.3	97.5 / 98.1
Carpet	68.4 / -	89.0 / 97.1	75.4 / 64.6	67.6 / 71.1	98.9 / 98.2	99.7 / 98.5	98.7 / 99.4	**100.0 / 99.7**
Grid	52.5 / -	52.0 / 85.7	61.3 / 60.3	57.6 / 45.4	**100.0** / 99.1	97.1 / 97.6	92.4 / 97.6	99.8 / **99.2**
Hazelnut	42.1 / -	43.2 / 95.8	67.0 / 59.7	69.4 / 64.1	99.6 / 98.5	**100.0** / 95.4	98.7 / 97.6	**100.0 / 99.7**
Leather	72.9 / -	94.7 / 97.2	81.0 / 66.9	45.8 / 31.5	**100.0** / 99.4	**100.0**/ 99.3	**100.0** / 99.0	**100.0 / 99.8**
Metal_nut	70.1 / -	74.6 / 79.9	90.2 / 62.7	60.9 / 60.7	97.2 / 86.3	99.7 / 77.0	97.2 / 98.4	**99.8 / 99.4**
Pill	72.8 / -	76.5 / 95.9	84.5 / 90.8	49.1 / 55.6	96.5 / **98.5**	94.6 / 97.3	89.1 / 96.1	**98.1 / 98.5**
Screw	58.0 / -	62.5 / 96.7	78.0 / 77.2	53.6 / 61.9	91.2 / **98.9**	**93.5** / 94.5	79.9 / 98.3	93.4 / 98.4
Tile	69.8 / -	71.0 / 72.6	86.8 / 63.3	66.9 / 33.0	98.6 / 94.2	**99.4** / 94.0	99.1 / 97.7	99.0 / **98.8**
Toothbrush	83.9 / -	80.0 / 95.3	95.3 / 91.1	50.6 / 34.9	99.2 / 94.1	**99.7 / 98.8**	98.3 / **98.8**	98.6 / **98.8**
Transistor	43.8 / -	45.8 / 90.8	72.1 / 61.4	61.5 / 48.3	88.5 / 87.6	**99.1** / 90.4	99.0 / 95.2	98.7 / **97.7**
Wood	54.3 / -	66.8 / 90.2	76.0 / 62.0	79.2 / 64.6	97.6 / 96.1	98.9 / 95.2	95.2 / 91.8	**100.0 / 98.0**
Zipper	75.9 / -	77.3 / 96.9	90.7 / 76.4	66.9 / 63.4	99.7 / 98.5	97.6 / 98.7	91.6 / 94.4	**99.9 / 99.2**
Average	65.5 / -	70.8 / 91.6	81.6 / 70.8	59.3 / 52.9	96.8 / 96.0	98.3 / 95.1	92.4 / 97.2	**98.8 / 98.6**

Table 8. Detailed results for MVTec AD in the *Overlap* setting.

Type	One-class classification					Fully unsupervised		
Method	CS-Flow [11]	PaDiM [3]	PatchCore [10]	SimpleNet [8]	RealNet [16]	SoftPatch [15]	InReaCh [9]	FUN-AD (*Ours*)
Candle	86.7 / -	91.2 / 99.4	95.0 / 99.2	93.9 / 97.9	**94.7 / 99.6**	93.8 / **99.6**	90.1 / 98.7	94.4 / 99.5
Capsules	68.2 / -	56.6 / 94.0	69.5 / 96.2	76.5 / 96.8	82.5 / 98.9	69.2 / 97.7	57.9 / 93.0	**93.8 / 99.5**
Cashew	86.1 / -	90.0 / 99.0	94.7 / 99.0	90.7 / 99.5	84.3 / 98.2	95.2 / 99.1	77.1 / 98.2	**96.6 / 99.7**
Chewinggum	93.8 / -	95.2 / 98.9	98.4 / 98.5	96.0 / 97.5	**99.8 / 99.9**	98.7 / 99.1	77.8 / 98.2	99.3 / 99.7
Fryum	78.0 / -	89.1 / 97.7	89.4 / 91.2	91.5 / 94.8	89.1 / 94.5	92.0 / 96.0	86.3 / 96.4	**96.9 / 98.2**
Macaroni1	81.4 / -	83.2 / 99.1	86.5 / 97.3	88.7 / 98.1	**98.1 / 99.9**	89.8 / 98.8	83.7 / 98.0	96.4 / **99.9**
Macaroni2	60.6 / -	60.4 / 95.6	64.8 / 89.3	72.1 / 94.6	**90.0 / 99.6**	57.6 / 95.1	57.4 / 95.8	87.0 / 99.1
PCB1	90.7 / -	94.2 / 99.6	93.0 / 86.5	93.0 / 94.8	93.8 / 99.4	95.1 / **99.8**	93.8 / 99.6	**96.2** / 99.6
PCB2	85.8 / -	91.8 / 99.2	**96.3** / 98.8	94.8 / 99.0	95.5 / 96.9	93.9 / **99.3**	91.9 / 98.7	91.2 / 97.8
PCB3	84.3 / -	85.7 / 99.1	93.8 / 96.9	95.2 / 99.2	**95.9** / 99.1	92.3 / **99.4**	93.3 / 99.3	91.4 / 98.6
PCB4	95.3 / -	97.1 / 98.2	98.0 / 96.3	97.8 / 96.1	98.9 / 98.9	99.2 / 99.1	**99.7 / 99.5**	97.6 / **99.5**
Pipe_fryum	77.3 / -	95.2 / 98.3	98.5 / 99.2	94.6 / 99.7	98.8 / 99.3	98.8 / 99.5	96.7 / **99.8**	**99.3 / 99.8**
Average	82.3 / -	85.8 / 98.3	89.8 / 95.7	90.4 / 96.7	93.5 / 98.7	89.6 / 98.5	83.8 / 97.6	**95.0 / 99.2**

Table 9. Detailed results for VisA in the *No overlap* setting.

Type	One-class classification					Fully unsupervised		
Method	CS-Flow [11]	PaDiM [3]	PatchCore [10]	SimpleNet [8]	RealNet [16]	SoftPatch [15]	InReaCh [9]	FUN-AD (*Ours*)
Candle	68.1 / -	79.7 / 99.1	85.3 / 87.5	47.6 / 49.0	**95.3 / 99.4**	93.7 / **99.4**	85.2 / 95.5	93.8 / **99.4**
Capsules	48.6 / -	45.3 / 78.9	65.2 / 83.9	50.3 / 50.2	82.6 / 96.8	70.5 / 89.2	53.1 / 89.0	**93.3 / 99.4**
Cashew	64.4 / -	72.0 / 85.2	86.8 / 85.7	61.4 / 60.8	88.9 / 96.6	94.1 / 98.9	75.5 / 91.0	**96.9 / 99.7**
Chewinggum	57.8 / -	76.2 / 82.1	87.9 / 78.1	51.9 / 59.9	**99.9** / 99.4	97.9 / 98.9	76.1 / 90.4	99.2 / **99.7**
Fryum	73.1 / -	71.3 / 92.9	84.3 / 82.5	51.9 / 80.0	86.1 / 94.8	92.9 / 92.4	78.1 / 94.7	**96.5 / 98.2**
Macaroni1	65.1 / -	68.8 / 97.5	80.2 / 84.0	45.7 / 49.3	**98.1 / 99.8**	89.0 / 97.2	75.5 / 88.1	96.7 / **99.8**
Macaroni2	48.2 / -	48.4 / 89.7	58.2 / 80.4	51.2 / 57.0	**88.8 / 99.2**	56.5 / 85.9	51.0 / 91.4	87.7 / **99.2**
PCB1	72.5 / -	77.4 / 98.0	87.3 / 71.4	44.7 / 60.8	93.2 / 98.9	95.9 / **99.6**	94.4 / 97.2	**96.7 / 99.6**
PCB2	68.7 / -	74.1 / 96.4	86.3 / 83.5	43.9 / 46.8	91.0 / 96.5	**93.0 / 98.0**	86.7 / 94.1	91.5 / **98.0**
PCB3	67.5 / -	69.6 / 97.0	85.6 / 76.6	53.7 / 65.8	90.0 / 96.3	**92.4 / 98.7**	83.1 / 97.3	91.7 / 98.2
PCB4	76.3 / -	82.1 / 95.7	93.9 / 83.0	46.9 / 68.0	97.0 / 96.6	**99.2** / 97.9	99.0 / 96.7	97.8 / **99.5**
Pipe_fryum	61.8 / -	76.2 / 95.7	87.1 / 82.6	56.2 / 47.2	98.1 / 97.7	98.4 / 99.4	84.4 / 97.5	**99.4 / 99.8**
Average	64.3 / -	70.1 / 92.3	82.3 / 81.6	50.4 / 57.9	92.4 / 97.7	89.5 / 96.3	78.5 / 93.6	**95.1 / 99.2**

Table 10. Detailed results for VisA in the *Overlap* setting.

References

[1] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. MVTec AD — A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9584–9592, 2019. 1, 2

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 9650–9660, 2021. 1, 2, 4

[3] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. PaDim: a patch distribution modeling framework for anomaly detection and localization. *ICPR 2020: 25th International Conference on Pattern Recognition Workshops and Challenges*, 12664:475–489, 2021. 5, 6

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1

[5] Yibin Huang, Congying Qiu, and Kui Yuan. Surface defect saliency of magnetic tile. *The Visual Computer*, 36:85–96, 2020. 2

[6] Jiin Im, Yongho Son, and Je Hyeong Hong. FUN-AD: Fully Unsupervised Learning for Anomaly Detection with Noisy Training Data, 2025. 1, 2

[7] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 1, 3

[8] Zhikang Liu, Yiming Zhou, Yuansheng Xu, and Zilei Wang. Simplenet: A simple network for image anomaly detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20402–20411, 2023. 1, 5, 6

[9] Declan McIntosh and Alexandra Branzan Albu. Inter-realization channels: Unsupervised anomaly detection beyond one-class classification. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 6285–6295, 2023. 2, 3, 5, 6

[10] 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)*, pages 14318–14328, 2022. 5, 6

[11] Marco Rudolph, Tom Wehrbein, Bodo Rosenhahn, and Bastian Wandt. Fully convolutional cross-scale-flows for image-based defect detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1088–1097, 2022. 5, 6

[12] Ozan Sener and Silvio Savarese. Active Learning for Convolutional Neural Networks: A Core-Set Approach. In *International Conference on Learning Representations*, 2018. 2

[13] Dong Wang and Xiaoyang Tan. Unsupervised feature learning with c-svddnet. *Pattern Recognition*, 60:473–485, 2016. 3

[14] Gaoang Wang, Yibing Zhan, Xinchao Wang, Mingli Song, and Klara Nahrstedt. Hierarchical semi-supervised contrastive learning for contamination-resistant anomaly detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 110–128, 2022. 3

[15] Jiang Xi, Jianlin Liu, Jinbao Wang, Qiang Nie, Kai WU, Yong Liu, Chengjie Wang, and Feng Zheng. SoftPatch: Unsupervised anomaly detection with noisy data. *Advances in Neural Information Processing Systems*, 35:15433–15445, 2022. 1, 2, 3, 5, 6

[16] Ximiao Zhang, Min Xu, and Xiuzhuang Zhou. RealNet: A Feature Selection Network with Realistic Synthetic Anomaly for Anomaly Detection. *arXiv preprint arXiv:2403.05897*, 2024. 5, 6