Beyond Academic Benchmarks: Critical Analysis and Best Practices for Visual Industrial Anomaly Detection

Supplementary Material

7. Appendices

The full results for all methods and categories can be found in this file: https://eu.mydrive.ch/shares/88698/f808671f908f1e93d2a5597c4cd29e9c/download/452956104-1740916171/results.json

- **Sec. 7.1: Models** - Implementation details for models.
- **Sec. 7.2: Datasets** - Implementation details for datasets.
- **Sec. 7.3: Synthetic data drift** - Detailed descriptions of synthetic perturbations used to create synthetic data drift.
- **Sec. 7.4: Score distributions** - Visualization of image-level score distributions.
- **Sec. 7.5: More results** - PG2 and PB2 image-level results for data drift and noisy labels experiments. Results for experiments per dataset.

7.1. Models

This subsection describes the details of the implementation of the models we use. All models use the pretrained feature extractor WideResNet-50-2 by TorchVision [47] unless stated otherwise. Input size is set to 256×256 for all experiments except Input Resolution. We remove center crop augmentation and report results for the last epoch without an early stop (unless it is a Validation Strategies experiment). The rest of the settings we use follow the original paper's settings.

PatchCore: Implementation by Anomalib [1]. The feature extractor uses layers 2 and 3. The coreset sampling ratio is 0.1, and the number of neighbors is 9. For larger datasets (such as VAD and one of the classes in BTAD), for resolution 512×512, we sample 25% of images for training to fit into GPU memory, similar to [3].

Reverse Distillation: Implementation by Anomalib [1]. Trained for 200 epochs.

CSFlow: Implementation by Anomalib [1]. EfficientNet B5 by TorchVision [47] as a feature extractor. Trained for 240 epochs.

MMR: Official implementation[1] re-implemented into Anomalib. Trained for 200 epochs with 50 warmup epochs.

MSFlow: Official implementation[2]. The original code calculates two anomaly maps, one using addition to calculate pixel-level AUROC and another using multiplication to calculate pixel-level AUPRO. We modified it to calculate only one anomaly map through the addition; the same anomaly map is used to calculate the anomaly score, according to the paper.

DRAEM: Official implementation[3]. Trained for 700 epochs.

SimpleNet: Official implementation[4] re-implemented into Anomalib. Trained for 160 epochs.

GLASS: Official implementation[5]. For VAD and BTAD, we do not use background masks because there is no background in the images. For the rest of the datasets, masks are created using SAM [33]. Trained for 640 epochs.

DRA: Official implementation [22]. This model calculates only image-level anomaly scores. We use the backbone ResNet-18 because other backbones have not been implemented. Trained for 30 epochs. We removed random rotation augmentation because it makes results for some datasets worse [2].

DevNet: Official implementation [45]. This model calculates pixel-level anomaly scores separately from image-level scores based on gradient back-propagation, we do not include this in our evaluation. Trained for 50 epochs.

7.2. Datasets

This subsection describes the details of the datasets we use. Datasets not mentioned below are used without any changes, similar to the original papers. For **MVTecLOCO**, we report mean results for logical and structural defects, similar to the original paper. For **AeBAD**, we report mean results for three types of data drift and results without data drift separately.

SensumSODF: The original paper uses three splits; we use one split to simplify evaluation.

Real-IAD: We use a version of the dataset with a shot from above only. Classes included into our subset: *pcb, phone_battery, plastic_nut, plastic_plug, porcelain_doll, terminalblock, usb, woodstick*.

VIADUCT: Classes included into our subset: *3_pole_socket_housing, cylinder_screw, damper_large, dsub_connector, mains_tester, pcb, retractor, ring_cable_lug, tack, terminal_block_a, threaded_fitting, valve_handle_blue*.

7.3. Synthetic data drift

To simulate data drift, an augmentation pipeline was created to synthetically add data drift to the images of the test set.

[1] https://github.com/zhangzilongc/MMR
[2] https://github.com/cool-xuan/msflow
[3] https://github.com/VitjanZ/DRAEM
[4] https://github.com/DonaldRR/SimpleNet
[5] https://github.com/cqylunlun/GLASS

Table 10. Overview of the transforms used in the data drift experiment. Each test set image was augmented with 1 or 2 transforms picked randomly from different categories.

Category	Transform	Parameter	Value/Range*
Motion /camera quality	Gaussian Blur	Kernel size	7
		Sigma	[0.1, 1.5]
	Gaussian Noise	Mean	0.5
		Standard Deviation	1.0
		Scale	[0.01, 0.05]
Lighting conditions	Color Jitter	Brightness factor	[0.5, 1.5]
		Contrast factor	[0.5, 1.5]
		Saturation factor	[0.5, 1.5]
	Random Shadow	Number of Shadow Layers	3
		Brightness factor	0
Camera position	Random Rotation	Rotation angle	[-5, 5]
	Random Cropping	Scale	[0.8, 1.0]
	Perspective Transform	Distortion scale	0.2

* The used parameter value was sampled uniformly from the specified range of values

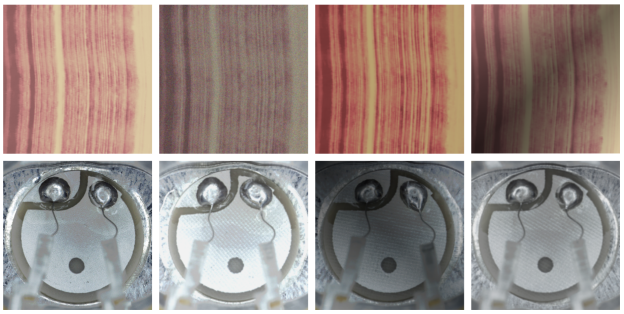

Figure 5. Synthetic perturbations. The image on the left in each row shows the original data, and the images on the right show different augmentations produced using our pipeline.

The data drift augmentations consisted of 7 types of transforms across 3 categories, listed in Table 10. Gaussian blur and Gaussian noise to simulate camera motion and camera quality, color jitter and random shadow to simulate varying lighting conditions, and random rotation, random cropping and perspective transforms to simulate varying camera placement conditions. A random selection of 1 or 2 data drift categories was applied to each image from the test set. Within each of the selected categories, a single transform type was chosen randomly.

7.4. Score distributions

7.5. More results

Due to restricted space in the paper, we include PG2 and PB2 image-level results for Noisy Labels and Data Drift experiments in Tab. 11 and Tab. 12 respectively. It can be seen that the best-performing models in terms of PG2 are different compared to image-level AUROC. In the Data Drift experiment, MSFlow performs better than PatchCore, which is also demonstrated in Fig. 6, which shows a clearer separation of good images (measured by PG2) by MSFlow versus PatchCore. With the PG2 metric, PatchCore clearly outperforms other models; it is visualized in Fig. 6.

Table 11. Noisy labels experiment, additional results. The best result is marked in bold. D denotes *the mean value for VAD, SensumSODF, and VIADUCTs datasets*. M means *Methods*. Metrics are im.PG2/im.PB2.

M	D	D4%	D8%	D16%
InR	15.9/28.2	18.7/60.4	18.3/61.2	19.1/58.8
PtC	37.0/56.9	30.0/87.1	33.1/81.4	26.8/79.5
RD	33.1/48.9	30.0/71.7	30.9/65.2	30.6/66.8
MMR	32.7/49.8	28.3/69.6	27.1/66.9	27.1/64.4
CSF	32.7/47.4	31.1/63.4	26.2/57.8	26.0/55.5
MSF	38.8/43.9	36.2/49.1	32.4/45.4	28.0/37.9
SN	25.7/42.9	18.5/44.1	15.8/38.6	14.0/28.7
DR	16.4/34.8	14.0/46.8	11.2/34.9	14.9/27.5

Table 12. Data drift experiment, additional results. The best result is marked in bold. D denotes *the mean value for RIADs, BTAD, VAD*. M means *Methods*. Metrics are im.PG2/im.PB2.

M	D	D+dr	AeBAD	AeBAD+dr
PtC	40.9/66.5	14.2/25.3	11.7/15.3	15.2/13.2
RD	44.7/63.8	2.0/0.8	38.7/22.2	31.7/16.3
MMR	44.4/63.5	17.3/5.4	26.1/42.0	23.7/27.8
CSF	37.1/54.3	3.6/4.1	9.1/5.9	6.9/3.3
MSF	38.4/41.2	5.2/0.0	10.9/0.0	3.8/0.0
SN	31.4/50.3	3.4/0.9	2.2/0.0	3.9/0.0
DR	28.2/43.1	3.4/2.9	7.8/0.0	4.4/0.0
GL	28.8/49.4	5.8/1.5	15.7/0.0	4.9/0.0

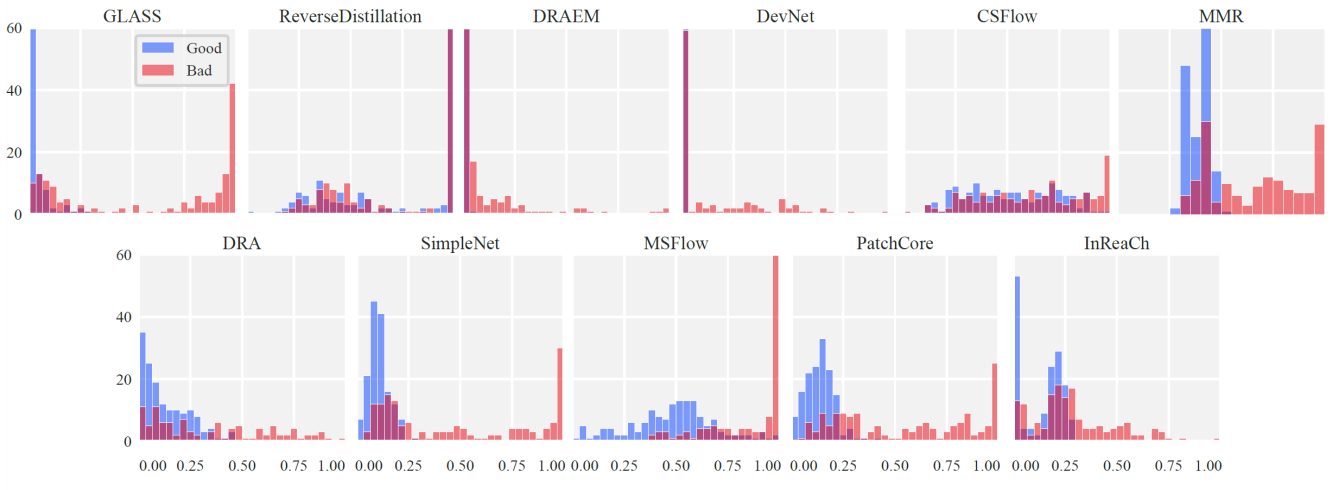

Figure 6. Image-level score distributions for different models for SensumSODF, capsule. Bad part scores are shown in red, and good in blue. The X axis shows scores, and the Y axis shows frequency.

Table 13. Noisy labels experiment per dataset. Metrics are im.AUROC/pix.AUPRO.

Method	Dataset	D4%	D8%	D16%
InR	SSODF	77.0/69.4	80.7/70.1	77.7/70.2
	VIADUCTs	76.6/85.4	75.3/83.9	75.2/84.7
	VAD	83.1/0.0	83.0/0.0	82.8/0.0
PtC	SSODF	92.6/91.0	92.6/90.9	90.7/89.7
	VIADUCTs	81.8/72.8	80.9/71.3	78.8/69.7
	VAD	87.4/0.0	85.6/0.0	84.8/0.0
RD	SSODF	91.0/93.4	91.1/93.7	89.4/93.8
	VIADUCTs	81.5/89.7	81.0/90.4	79.7/90.2
	VAD	83.8/0.0	82.8/0.0	82.9/0.0
MMR	SSODF	90.8/93.1	90.1/93.4	89.6/94.5
	VIADUCTs	79.5/87.3	78.8/87.2	78.1/86.8
	VAD	84.3/0.0	83.3/0.0	82.1/0.0
CSF	SSODF	89.5/40.0	88.1/42.4	88.0/44.4
	VIADUCTs	81.1/34.6	78.7/33.6	76.5/35.2
	VAD	80.3/0.0	78.8/0.0	78.0/0.0
MSF	SSODF	90.1/82.0	88.9/84.8	86.9/87.9
	VIADUCTs	82.9/74.1	80.9/74.8	78.4/75.0
	VAD	83.5/0.0	82.2/0.0	80.4/0.0
SN	SSODF	80.0/71.8	74.1/60.9	70.0/56.9
	VIADUCTs	84.1/86.7	81.3/86.0	79.9/84.1
	VAD	64.3/0.0	61.2/0.0	56.6/0.0
DR	SSODF	83.9/66.7	82.1/60.4	78.6/62.5
	Viaducts	76.4/63.4	75.9/61.5	71.4/60.2
	Vad	54.9/0.0	57.2/0.0	55.8/0.0

Table 14. Input size experiment per dataset. Metrics are im.AUROC/pix.F1Max/im.PG2.

Method	Dataset	Size 128	Size 512
PtC	RIADs	83.4/18.9/15.9	94.4/46.7/56.6
	BTech	93.1/41.7/50.9	95.9/64.7/75.5
	VAD	79.4/0.0/11.8	88.7/0.0/21.0
RD	RIADs	86.6/24.2/27.7	94.9/51.3/51.1
	BTech	93.2/51.3/64.5	82.8/60.6/64.0
	VAD	81.8/0.0/12.9	69.8/0.0/8.8
MMR	RIADs	78.8/24.7/10.4	94.9/51.0/50.5
	BTech	86.6/41.8/38.9	92.8/54.8/50.9
	VAD	79.7/0.0/10.2	78.9/0.0/10.4
CSF	RIADs	77.2/1.3/12.2	89.2/4.8/36.5
	BTech	91.5/23.0/41.0	95.0/29.4/71.1
	VAD	76.1/0.0/12.2	84.4/0.0/24.3
MSF	RIADs	81.3/1.4/17.9	91.5/22.8/42.2
	BTech	85.6/17.0/58.4	90.8/44.0/64.1
	VAD	76.7/0.0/10.7	75.6/0.0/14.5
SN	RIADs	77.4/16.6/11.4	94.2/40.7/54.2
	BTech	91.9/37.7/61.3	87.5/38.5/41.3
	VAD	61.0/0.0/4.9	61.7/0.0/3.9
DR	RIADs	82.5/53.5/23.4	84.6/45.5/19.3
	BTech	88.9/31.6/49.8	90.8/24.4/23.9
	VAD	70.1/0.0/7.2	59.3/0.0/4.3
GL	RIADs	82.9/26.9/18.3	92.1/55.1/36.2
	BTech	74.5/29.6/32.1	95.2/51.0/62.1
	VAD	74.8/0.0/11.4	80.3/0.0/12.8

Another interesting outcome in the Noisy Labels experiment is that adding a little label contamination (4%) strongly improves the separation of bad images (PB2) for almost all models. PatchCore shows the best improvement at 30.2 points, ReverseDistillation improves by 22.8 points. This phenomenon requires further investigation. It also demonstrates the importance of metrics, which qualify the classification results differently for good and bad parts.

Results per dataset are in Tab. 13, Tab. 14 and Tab. 15. Tab. 14 gives a particular insight into how input size is connected to the size of defects presented in the dataset. RIADs show improvement for all models, demonstrating that a bigger input size helps to detect small defects. Meanwhile, VAD and BTech show no improvement or even reduction due to some large defects not processed properly by the feature extractor.

Table 15. Data drift experiment per dataset. Metrics are im.AUROC/pix.AUPRO/im.PG2.

Method	Dataset	No drift	Drift
PtC	RIADs	91.4/92.0/39.0	65.9/19.5/0.7
	BTech	95.5/76.9/67.3	87.8/40.4/28.8
	VAD	88.0/0.0/16.5	69.4/0.0/13.0
RD	RIADs	93.2/95.0/46.3	47.9/25.5/0.0
	BTech	94.3/79.5/67.7	46.2/11.4/4.1
	VAD	84.7/0.0/20.1	51.7/0.0/2.0
MMR	RIADs	92.4/96.3/42.7	67.2/15.4/16.7
	BTech	93.7/77.9/62.9	67.0/26.7/22.3
	VAD	87.6/0.0/27.6	61.3/0.0/12.8
CSF	RIADs	86.3/47.9/22.7	57.9/19.0/0.0
	BTech	95.1/57.5/71.5	67.9/12.4/8.7
	VAD	82.2/0.0/17.1	52.4/0.0/2.0
MSF	RIADs	89.2/85.6/31.1	55.7/7.1/6.7
	BTech	90.0/62.6/57.4	51.9/0.0/5.9
	VAD	84.4/0.0/26.6	53.1/0.0/7.2
DR	RIADs	85.6/86.8/26.9	52.4/16.4/7.2
	BTech	89.6/52.6/55.9	53.7/9.6/4.8
	VAD	57.7/0.0/1.8	51.2/0.0/3.1
GL	RIADs	88.8/67.1/30.7	56.1/15.8/5.8
	BTech	90.9/61.1/45.5	53.6/7.7/6.4
	VAD	79.2/0.0/10.3	58.1/0.0/5.1
SN	RIADs	82.6/77.6/35.4	55.1/15.8/0.0
	BTech	89.7/68.0/53.5	57.3/15.4/6.1
	VAD	69.7/0.0/5.3	57.1/0.0/4.1