Detecting visual anomalies in an industrial environment: Unsupervised methods put to the test on the AutoVI dataset

Philippe Carvalho, Meriem Lafou, Alexandre Durupt, Antoine Leblanc, Yves Grandvalet

Computers in Industry 163 (2024) 104151

Contents lists available at ScienceDirect

Computers in Industry

journal homepage: www.sciencedirect.com/journal/computers-in-industry

Detecting visual anomalies in an industrial environment: Unsupervised methods put to the test on the AutoVI dataset

Philippe Carvalho [a], Meriem Lafou [b], Alexandre Durupt [a,*], Antoine Leblanc [b], Yves Grandvalet [c]

[a] Université de technologie de Compiègne, Roberval, rue du docteur Schweitzer, CS 60319, 60203 Compiègne Cedex, France
[b] Renault Group, Renault Technocentre, 1 avenue du Golf, 78084 Guyancourt, France
[c] Université de technologie de Compiègne, Heudiasyc, UMR CNRS 7253, 57 avenue de Landshut, CS 60319, 60203 Compiègne Cedex, France

ARTICLE INFO

Dataset link: https://zenodo.org/records/10459003, https://github.com/phcarval/autovi-paper-code, https://github.com/phcarval/autovi_evaluation_code, https://github.com/openvinotoolkit/anomalib/, https://www.mvtec.com/company/research/datasets/mvtec-loco

Keywords:
Unsupervised defect detection
Assembly lines
Machine learning
Visual inspection
Industry 4.0

ABSTRACT

The methods for unsupervised visual inspection use algorithms that are developed, trained and evaluated on publicly available datasets. However, these datasets do not reflect genuine industrial conditions, and thus current methods are not evaluated in real-world industrial production contexts. To answer this shortcoming, we introduce AutoVI, an industrial dataset of visual defects that can be encountered on automotive assembly lines. This dataset, comprising six inspection tasks, was designed as a benchmark to assess the performance of defect detection methods under realistic acquisition conditions. We analyze the performance of current state-of-the-art methods and discuss the difficulties specifically encountered in the industrial context. Our results show that current methods leave considerable room for improvement. We make AutoVI publicly available to develop unsupervised detection methods that will be better suited to real industrial tasks.

1. Introduction

Research in the area of defect detection and visual inspection has garnered very significant interest in recent times (Gao et al., 2021; Lindemann et al., 2021; Pang et al., 2021; Zeiser et al., 2023). Visual anomalies can be described in terms of structural and logical defects (Bergmann et al., 2022). Structural defects are found in the local visual structure of an object or texture, such as scratches, burns, impacts, or spots. Logical defects refer to problems stemming from the incorrect visual arrangement of correct objects, such as cables that are connected to the wrong clamps.

The vast majority of algorithms used today for visual inspection are based on data-driven deep learning algorithms. Deep learning algorithms usually require large amounts of training data to work, but offer much better performance than traditional algorithms (Krizhevsky et al., 2017). Applying these methods for visual inspection in the manufacturing industry leads to a major drawback: the need to constitute expensive datasets to train and evaluate the models. Such datasets are especially costly to make: a significant number of images must be taken of both defective and non-defective items. Labeling the data is a time-consuming task that requires a field expert. Besides, collecting defective items is tedious as most industrial production systems yield an extremely low rate of defects.

That being said, new research interests lie in the development of unsupervised methods. These methods are able to identify defects while having only used defect-free images for training. These methods carry several powerful advantages compared to methods that require defective items for training, known as supervised methods.

- They are able to train using only non-defective items, enabling them to be implemented without collecting defective items, which can be difficult to acquire as they have a much lower prevalence than non-defective items;
- They can be more easily implemented in a new production line as there are only relatively few non-defective items to collect for training;
- They can be used to detect all kinds of defects, not only those that are collected in the defect library.

Nonetheless, these deep learning methods still require training and testing data to be developed and compared. Currently available datasets do not necessarily represent genuine industrial production conditions (Božič et al., 2021; Defard et al., 2021; Roth et al., 2022; Zavrtanik et al., 2021a, 2022; Zhang et al., 2023). Hence, newly developed algorithms are designed and evaluated for their ability to

* Corresponding author.
E-mail addresses: philippe.carvalho@utc.fr (P. Carvalho), meriem.lafou@renault.com (M. Lafou), alexandre.durupt@utc.fr (A. Durupt), antoine.leblanc@renault.com (A. Leblanc), yves.grandvalet@utc.fr (Y. Grandvalet).

https://doi.org/10.1016/j.compind.2024.104151
Received 22 February 2024; Received in revised form 21 May 2024; Accepted 20 August 2024
Available online 2 September 2024

Table 1
Summary of the main features of the described datasets.

Dataset	N. images	N. defects	N. classes	Shooting conditions	Logical defects
DAGM (Wieler et al., 2007)	16100	2100	10	Artificial	N
NEU (Song and Yan, 2013)	1800	1800	6	Laboratory	N
Severstal Steel (Severstal, 2019)	12572	Not given	1	Industrial (steel)	N
MVTec AD (Bergmann et al., 2019)	5354	1258	15	Laboratory	Y[a]
KolektorSDD (Tabernik et al., 2020)	399	52	1	Industrial (plastic)	N
KolektorSDD2 (Božič et al., 2021)	3335	356	1	Industrial (unspecified)	N
MVTec LOCO (Bergmann et al., 2022)	3644	993	5	Laboratory	Y
VisA (Zou et al., 2022)	10821	1200	12	Laboratory	N
AutoVI (Ours)	3950	887	6	Industrial (assembly)	Y

[a] Bergmann et al. (2022) indicate the presence of logical defects in MVTec AD that were not discussed in Bergmann et al. (2019, 2021).

improve performance on datasets that do not represent real-world production contexts. This poses two problems. Firstly, we have no public datasets available for reproducible evaluation of algorithms in real industrial contexts. Secondly, the development and publication of new algorithms is biased by the datasets publicly available, in that they favor methods that are adapted to these datasets but not necessarily to real production environments. Therefore, a new public, real-world, industrial dataset would provide:

- a reproducible benchmark of state-of-the-art algorithms on publicly available genuine industrial data;
- a challenging and genuine dataset that promotes the development of more powerful machine learning methods that can be successfully applied in an industrial setting.

In this paper, we propose a detailed benchmark of several state-of-the-art methods on a new public dataset, the Automotive Visual Inspection Dataset (AutoVI). This dataset has been built on the assembly lines of a major automotive group. It contains images of several inspection tasks, ranging from part inspection to cable connection checking, with both non-defective and defective items, specifically created on the assembly line to evaluate visual inspection algorithms. Our contributions are as follows:

- We survey existing unsupervised defect detection methods for anomaly detection and existing datasets for industrial visual inspection;
- We propose a new publicly available dataset, the Automotive Visual Inspection Dataset (AutoVI), as a genuine industrial dataset to be used as a reference for benchmarking and developing future unsupervised defect detection methods;
- We test on this dataset the state-of-the-art methods for detection and segmentation of anomalies;
- We analyze the performance of these methods and provide recommendations for their usage and improvement.

2. Related works

We start by discussing existing public datasets related to visual defect detection in production line environments. We expand our analysis to include datasets related to general manufacturing processes. We show that there are no existing datasets which are captured in a genuine industrial environment, thus motivating the release of AutoVI.

We follow by discussing the characteristics of existing unsupervised defect detection methods for defect detection in order to select a representative subset of algorithms to be benchmarked on AutoVI. We exclusively focus on public datasets and methods that were openly published and tested on publicly accessible datasets.

2.1. Datasets

Since 2007, a number of datasets have been published with the aim to mimic to some extent the visual characteristics of industrial defect detection problems. We list datasets that correspond to industrial production scenarios, i.e. textures, objects produced industrially. Fig. 1 shows representative samples of all mentioned datasets, while Table 1 shows the main characteristics of each dataset: the number of images, defects and classes, the shooting conditions (artificial, laboratory or industrial), and the presence of logical defects. We list here these datasets in chronological order of publication.

The DAGM dataset. This dataset (see Fig. 1(a)) was published in 2007 (Wieler et al., 2007) as a competition organized by the German Association for Pattern Recognition (Deutsche Arbeitsgemeinschaft für Mustererkennung e.V.) It contains 10 texture categories and a total of 16100 images including 2100 images of defective items. Of the 10 categories, 6 each contain 1000 defect-free images and 150 defect images, and the remaining 4 categories each contain 2000 defect-free images and 300 defect images. This dataset has been extensively used for the development of defect detection algorithms (Carvalho et al., 2022; Rački et al., 2018; Wang et al., 2018; Weimer et al., 2016). However, this dataset now features two significative limitations. Firstly, the images were artificially generated, whereas datasets made of real images are now publicly available. Secondly, the difficulty of the problem no longer presents a challenge, as Božič et al. (2021) correctly classifies all images using a mixed-supervision algorithm.

The NEU defect database. This dataset (see Fig. 1(b)), published in 2013, shows defective steel sheets (Song and Yan, 2013). It consists of six categories of 300 images each, each category showing a different kind of defect (crazing, pitting, scratches, etc.) for a total of 1800 images. This dataset, which is among the first public datasets showing real-world defects in a simulated manufacturing environment, was used as a benchmark in some publications (Božič et al., 2021; Cohn and Holm, 2021). However, this dataset only features defective items, and thus does not make it readily usable for unsupervised visual anomaly detection. Furthermore, it has not been created in a genuine manufacturing environment, and does not feature logical defects.

The severstal steel defect dataset. This dataset (see Fig. 1(c)) was published in 2018 as a Kaggle challenge (Severstal, 2019). It contains 12572 images of defective steel sheets and grids, which makes it the largest real-world manufacturing dataset to this day. As for the NEU defect dataset, it was used for some benchmarks (Božič et al., 2021) although little is known about the production context of the dataset (Carvalho et al., 2023). While this dataset has been created in a real-world production line, it does not feature logical defects, and only features steel sheets and grids, arranged into a single category.

The MVTec anomaly detection dataset (MVTec AD). This dataset (see Fig. 1(d)), published in 2019, shows 15 diverse categories of items (Bergmann et al., 2019, 2021). This is the first dataset that shows different categories of items and textures that mimics a realistic industrial inspection scenario. It contains 5354 images of industrially produced objects, such as screws, cables, bottles or pills, as well as images of items rarely found on industrial production lines, such as hazelnuts. It also includes textures, such as wood, metal, or cloth. Furthermore, this dataset encourages usage of unsupervised methods for training, as

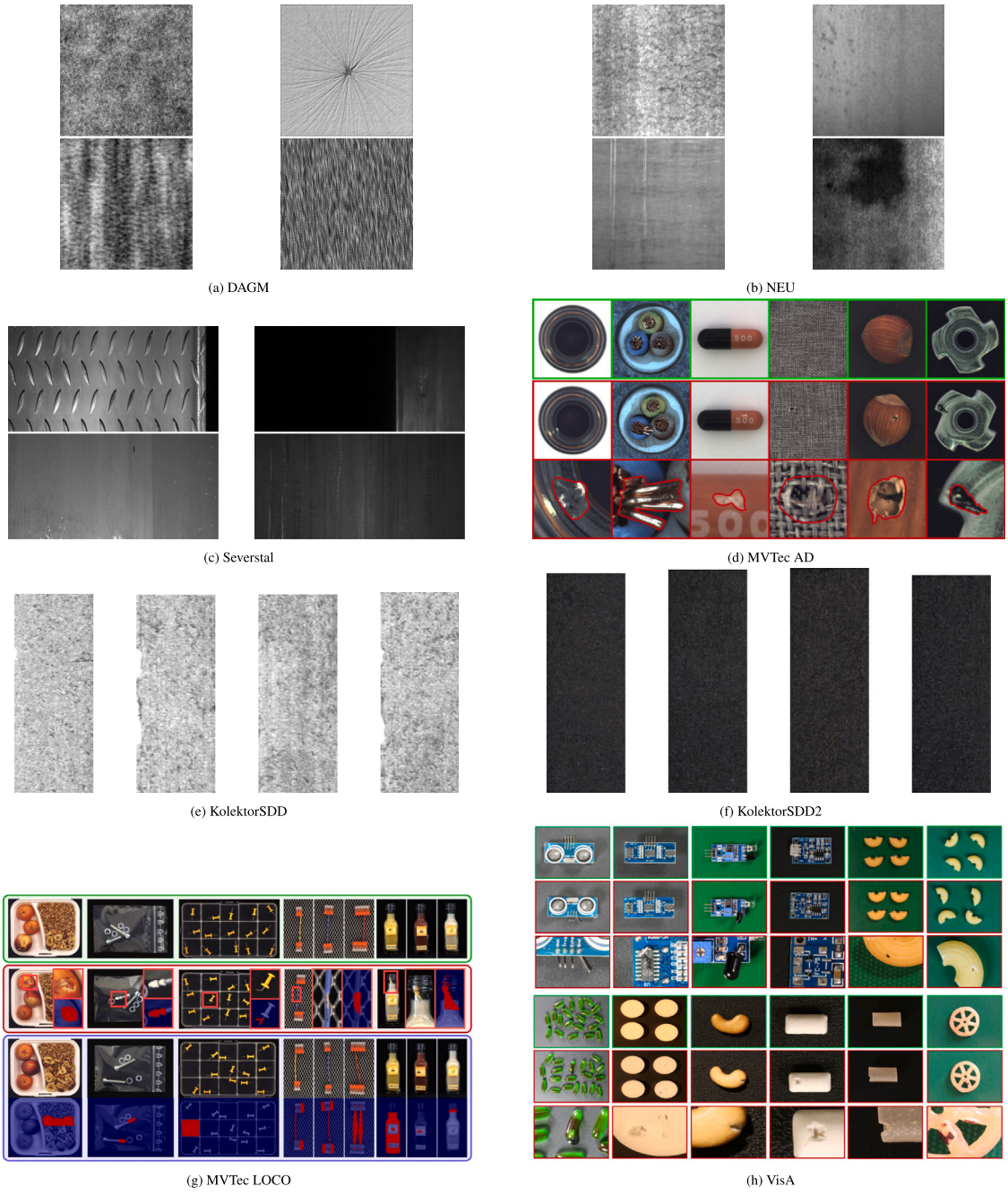


Fig. 1. Sample images from surveyed datasets.

it only includes defect images in the test set. This dataset has been extensively used as a benchmark (Batzner et al., 2024; Defard et al., 2021; Roth et al., 2022; Shi et al., 2023; Zavrtanik et al., 2022; Zhang et al., 2023). Although this dataset shows logical defects (Bergmann et al., 2022), they are not considered as a specific category in the dataset.

The KolektorSDD dataset. This dataset (see Fig. 1(e)) contains 399 images of plastic embeddings of electrical communators and was published in 2019 (Tabernik et al., 2020). It shows 347 images without defect and 52 images with defect. This dataset is the first to use images captured in a controlled industrial production environment. However, the value of this benchmark is limited by the small number of images of a single type of object. Furthermore, this dataset does not define a fixed train/test split, which does not restrain the use of this dataset for unsupervised methods. Finally, this dataset only proposes structural defects. For these reasons, this dataset has not been widely used as a reference for experimental studies in unsupervised anomaly detection.

The KolektorSDD2 dataset. This dataset (see Fig. 1(f)) contains 3335 images and was published in 2021 (Božič et al., 2021). This dataset contains 356 defective items and 2979 defect-free images. Similarly to the KolektorSDD dataset, these images are captured in a controlled industrial production environment. This dataset is a direct improvement over the first version, as it contains more images, RGB channels, and a fixed train/test categorization. However, this dataset still includes defective items in the training set, and does not include logical defects. As for the first version, this dataset has not been extensively used as a benchmark for unsupervised anomaly detection.

The MVTec logical anomalies dataset (MVTec LOCO). This dataset (see Fig. 1(g)), published in 2022, shows 5 categories of industrially-produced items (Bergmann et al., 2022). This dataset contains 3644 images, including 1568 images for testing, of which 993 images show defective items. MVTec LOCO, similarly to MVTec AD, simulates a real-world industrial inspection scenario in a controlled, non-industrial environment, with diverse industrially-produced items: breakfast boxes, pushpins, screw bags, juice bottles and splicing connectors. This dataset is the very first to explicitly introduce the concept of *logical defects*, compared to *structural defects*. While the latter refer to defects in the local structure of an object (namely, visual texture anomalies such as scratches, crazings, color spots, etc.), the former refer to defects in the global structure of the object. Examples in the bottom row of Fig. 1(g) show logical defects: from left to right, the wrong amount of cereal in the breakfast box, two long screws instead of one short and one long screw in the screw bag, a missing pushpin, wrong connector lengths and incorrectly connected cable, and wrongly labeled juice boxes. Generally, this category of defects refers to structures that are not defective by themselves – the longer screw in the previous example may not show any defect by itself. Rather, the defect is characterized by the presence of structurally correct elements in the wrong context. This dataset is the first to encourage the development of methods able to detect *logical defects*, over several industrially-produced categories of items, and including a fixed testing benchmark with both structural and logical defects. MVTec LOCO has been used as a benchmark in several research papers for unsupervised visual inspection (Batzner et al., 2024; Guo et al., 2023; Yao et al., 2023; Zhang et al., 2024).

The visual anomaly dataset (VisA). This dataset (see Fig. 1(h)) contains industrially produced items and was published in 2022 (Zou et al., 2022). With 10821 images, including 1200 anomalous samples, this dataset is the second-largest dataset for industrial visual inspection. It contains 12 categories, 4 of which showing circuit boards, the rest showing diverse items such as macaronis, candles, chewing-gums, cashews and fryums, a type of snack food. While the number of images is high, the defects do not show any logical defects. Pictures are captured in a controlled, non-industrial environment. This dataset has been used as a benchmark in a number of publications (Batzner et al., 2024; Jeong et al., 2023).

Summary. Existing datasets present a notable diversity of objects and textures for anomaly detection on industrial production lines. However, as noted in Table 1, there are very few datasets captured under genuine industrial conditions; of these, all focus on texture defects. To this day, no dataset dedicated to visual anomaly detection is made up of images captured on industrial production lines. The shooting conditions in these environments are very different from controlled laboratory conditions. It is impossible to exert complete control over environmental conditions: lighting, vibration, positioning of the object relative to the camera. Furthermore, large variations of the surrounding scene can be observed. We consider MVTec AD, MVTec LOCO and VisA to be well suited to benchmarking due to the large number of classes, representing a wide range of inspection tasks. Datasets that contain only a single class are not sufficiently informative as a benchmark, which needs to include a variety of different situations. Therefore, we exclude the DAGM dataset, which is artificial and only includes texture data, and the NEU dataset as it is not directly geared towards anomaly detection, only containing defective items. However, all mentioned datasets were acquired under laboratory conditions. Our dataset of images acquired in an industrial context allows the comparison of methods under conditions corresponding to the realities of industry.

2.2. Methods

A number of methods have been developed for visual defect detection and benchmarked on the public datasets described above. Such methods follow different design principles, that we describe according to the following categories: flow-based methods, reconstruction-based methods, patch-based methods or student-teacher methods.

Flow-based methods. Flow-based architectures rely on the concept of normalizing flows. Normalizing flows are a series of bijective transformations that are used to model a complex probability distribution using a simpler base distribution (Kobyzev et al., 2021; Papamakarios et al., 2021). For defect detection, normalizing flows are used to estimate the defect density on the image. Gudovskiy et al. (2022) introduce **CFlow** by using normalizing flows in a multi-scale pooling architecture in order to transform the distribution of anomaly-free patches into a Gaussian distribution. They generalize prior normalizing flow architectures (Dinh et al., 2017) and give a detailed theoretical study on the usage of normalizing flows to estimate the likelihoods of any distribution. Rudolph et al. (2022) introduce **CS-Flow** by using normalizing flows at different scales and using cross-scale flows, by making flows of different scales interact with each other.

Reconstruction-based methods. Reconstruction-based methods reconstruct the input image, usually using a GAN (Goodfellow et al., 2020) or an autoencoder (Goodfellow et al., 2016, Chapter 14), and compare the generated image to the original one in order to identify the defective region. Schlegl et al. (2019) introduce **f-AnoGAN**, which leverages the generator and discriminator from the GAN architecture while adding an encoder. This allows f-AnoGAN to function as an autoencoder, as it reconstructs the original image using the encoder and generator, and as a GAN, by comparing the reconstructed image with the original image. Akcay et al. (2019) introduce **GANomaly**, an architecture comprising two encoders, a decoder and a discriminator. Similarly to Schlegl et al. (2019), the input image is reconstructed and compared with its reconstruction. The presence of a second encoder is used to compare the latent spaces generated after encoding the original image and the reconstructed image. Zavrtanik et al. (2021b) introduce **RIAD**, an autoencoder-based method trained to reconstruct input images by inpainting. Input images are divided in a grid that is used to generate several images with randomly blackened cells. The autoencoder reconstructs each image's blackened areas, which are then assembled together and compared with the input image. Zavrtanik et al. (2021a) then propose an improvement of RIAD with **DRAEM**. DRAEM generates defects on training images using Perlin noise (Perlin, 1985). An autoencoder is then trained to reconstruct the corresponding defect-free image, while a second autoencoder is used to output a segmentation map of the defective area in the input image using the reconstructed image. Zavrtanik et al. (2022) also propose **DSR**, which again improves on DRAEM by proposing an architecture based on discretized latent space representations (Razavi et al., 2019). The defects are generated using Perlin noise directly in the quantized representations of the input image, the defective areas being replaced by quantized representations gathered from a dictionary defined on ImageNet. This algorithm leads to higher quality defects being generated, that are harder to detect.

Patch-based methods. Patch-based methods refer to architectures that break down the input image into smaller patches which are then used to identify the defective regions. Usually, such methods leverage a network trained on a larger dataset, such as ImageNet (Russakovsky et al., 2015). Rippel et al. (2020) has shown that using pre-trained features from a larger dataset increases performance for a specific anomaly detection task on the MVTec AD dataset. Namely, **Patchcore** (Roth et al., 2022) uses an architecture based on pretrained residual networks (ResNets) (He et al., 2015), where input image patches are fed to the pretrained network and intermediate activations are then used to populate a memory bank of features. These features are then used during inference to identify the defective patches by measuring their distance to the recorded patches. **PaDiM** (Defard et al., 2021) is a method that models each patch position by a multivariate Gaussian distribution calculated on the network activations output by all training patches.

Student-teacher methods. Some architectures make use of student-teacher methods, formed from the knowledge distillation framework (Hinton et al., 2015; Wang and Yoon, 2022). Bergmann et al. (2020) present **Uninformed Students (US)**, a student-teacher model for anomaly detection. An ensemble of student models is trained to mimic the outputs of a single teacher model. When evaluating on potentially defective items, the distance between the teacher's output and the students' outputs is used to identify the defective items. Bergmann et al. (2022) then introduce **GCAD**, a method that is specifically designed to identify both *structural* and *logical* defects as introduced in the MVTec LOCO dataset. GCAD uses two branches, one designed to detect local anomalies and another for global anomalies. The local branch extracts patch descriptors using knowledge distilled from a pretrained ResNet on ImageNet. The global branch uses knowledge distilled from the local model and uses an autoencoder-based architecture to detect errors in the global structure of the image. The global branch is therefore more suited to detect *logical* errors, while the local branch, which acts both as a student and a teacher, is more suitable to detect *structural* errors. Batzner et al. (2024) introduce **EfficientAD**, an architecture that uses a mixture of student-teacher and autoencoder models. The motivation is that, when trained on defect-free data, the single student model fails to reproduce the output of the single teacher network over defective local patterns, but is able to reproduce the global structure of the input image, which makes it suitable to detect local, structural errors. On the other hand, the autoencoder fails to reproduce defective global features, making it suitable for detecting large-scale logical errors.

Conclusion. There exists a great variety of methods for unsupervised visual anomaly detection. The majority of these methods has been developed for structural defect detection, except for GCAD and EfficientAD. We review the performance of these methods on the MVTec AD, MVTec LOCO and VisA datasets in Section 2.3. They were established to be the datasets best suited for benchmarking in the conclusion of Section 2.1, due to the diversity of available classes. We will identify the best-performing methods on these datasets to carry out our benchmark as described in Section 4 by training and testing these methods on the AutoVI dataset.

2.3. Performance of methods on mvtec AD and MVTec LOCO

We recall in Table 2 the results previously obtained by the methods reviewed in Section 2.2. They attain very high results on MVTec AD, with EfficientAD reaching an almost perfect Area Under the Receiver Operating Curve (AUROC) of 99.1. For MVTec LOCO, EfficientAD reaches an AUROC of 90.7, showing that logical defects offer a significant challenge for current state-of-the-art methods.

While the reported performances are impressive, they are obtained on images of excellent quality in a controlled environment. It remains to be seen how these methods perform under real production conditions. To this end, we present in the next section the AutoVI dataset, produced under real industrial conditions on automotive assembly lines. We will then evaluate the best methods on these data.

3. Presentation of the automotive visual inspection dataset

The datasets that are publicly available today are benchmarks for new defect detection methods. These datasets, most notably MVTec AD, MVTec LOCO and VisA, show a wide range of detection tasks on different objects and textures, and include a variety of defects for testing. However, these datasets benefit from controlled laboratory conditions. For example, Bergmann et al. (2021) have explicitly developed MVTec LOCO as an imitation of real industrial inspection scenarios, but the photographs are captured with complete control of the camera, the subject and the environment. Inspection conditions on large-scale production lines are hardly controlled, either because a closed control environment is too expensive to install or the inspection task itself cannot be carried out in a closed environment.

Table 2
Overview of previously reported performance, measured by average AUROC, of state-of-the-art methods on MVTec AD, MVTec LOCO and VisA datasets. Figures are taken from the original papers, except those marked with an asterisk (*), which are taken from the Papers with Code benchmark (https://paperswithcode.com/sota/anomaly-detection-on-mvtec-loco-ad and https://paperswithcode.com/sota/anomaly-detection-on-visa). Best results are indicated in **bold**.

	MVTec AD	MVTec LOCO	VisA
CFlow	98.2	–	91.5*
CS-Flow	98.7	–	–
f-AnoGAN	–	64.2*	–
GANomaly	–	–	–
RIAD	91.7	–	–
DRAEM	98.0	73.6*	–
DSR	98.2	82.6*	–
Patchcore	99.0	80.3*	–
PaDiM	97.9	–	–
US	86.0	–	–
GCAD	93.1	83.3	89.1*
EfficientAD	**99.1**	**90.7**	**98.1**

Table 3
Statistical overview of the AutoVI dataset with the number of training images, test images without defects (Test OK), test images with defects (Test $\overline{\text{OK}}$).

Category	Image size	Train	Test OK	Test $\overline{\text{OK}}$	Total
engine_wiring	(400 × 400)	285	285	322	892
pipe_clip	(400 × 400)	195	196	141	532
pipe_staple	(400 × 400)	198	199	127	524
tank_screw	(1000 × 750)	318	318	95	731
underbody_pipes	(1000 × 750)	161	161	184	506
underbody_screw	(1000 × 750)	373	374	18	765
Total		1530	1533	887	3950

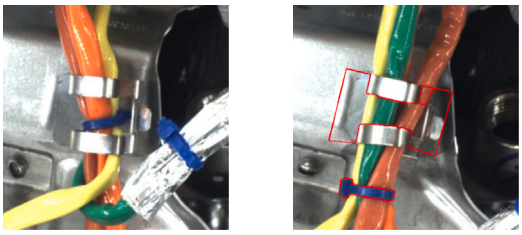
To the best of our knowledge, there are currently no public datasets captured on genuine industrial production lines. We propose the Automotive Visual Defect Dataset (AutoVI), a genuine industrial inspection dataset captured on the assembly lines of a major automotive group.

3.1. Description

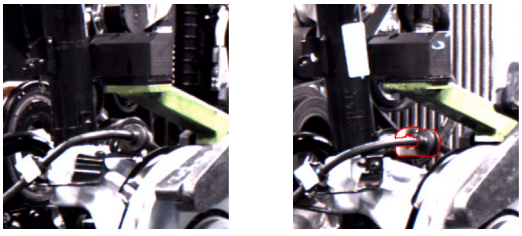
The AutoVI dataset consists of six classes corresponding to real inspection scenarios on automotive assembly lines. Samples for all categories can be viewed in Fig. 2 and the statistical overview of the dataset is given in Table 3.

Images were taken on the assembly lines of a single factory. The cameras were set up at varying distances from the object depending on the object photographed and the object's environment, so that they could be installed without blocking the movement of operators around the inspection area. The cameras stayed at a fixed location for the duration of the shooting, and only the items moved on the assembly line in front of the camera. Different lenses were used for the photographs. All categories but *engine_wiring* were photographed using 50 mm-long focals, due to them showing large objects, while *engine_wiring* was photographed with a 25 mm focal, due to the photographed area being smaller.

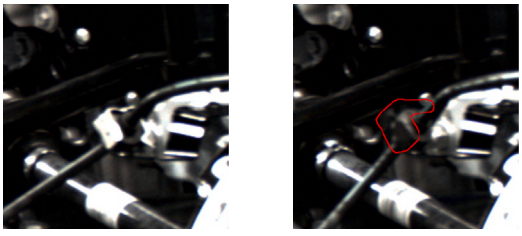
Due to the assembly line environment, it is impossible to control the environmental conditions, such as lighting, vibration, or the movement of the object. These conditions are interesting since they are representative of real viewing conditions on an assembly line, especially when dealing with large products, such as automobiles. Pictures that were overly difficult to classify – overly blurry, excessive amount of noise, item too far away from the average position – were removed. Test pictures always show elements that are present in the training pictures – besides the defective part itself –, so as to ensure that there are no false positives that correspond to new unlabeled elements only present in the test pictures. Pictures are available in the order of shooting



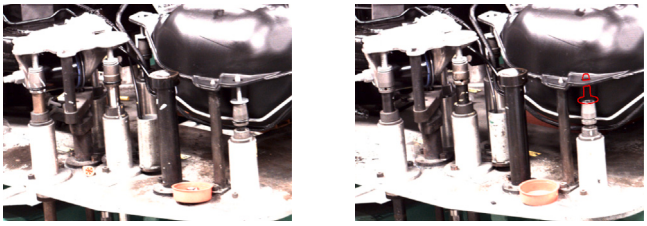
(a) *engine_wiring* (Wiring): the right image shows three defects (absent blue hoop, wrongly placed blue hoop and unfastened cable)



(b) *pipe_clip* (Clip): the right image shows the misplaced clip

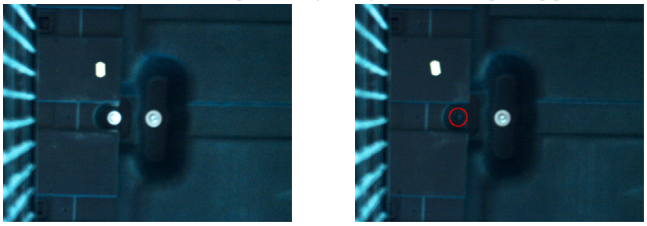


(c) *pipe_staple* (Staple): the right image shows the absent staple



(d) *tank_screw* (T. Screw): the right image shows the absent screw

(e) *underbody_pipes* (Pipes): the right image shows the misplaced pipes



(f) *underbody_screw* (U. Screw): the right image shows the missing screw

Fig. 2. Example of defective and non-defective items for each category of the AutoVI dataset.

within a single production series, the first pictures being assigned to the training set. This is done to better imitate the context of real-world assembly lines, which may exhibit some variation over the course of production.

Original images for all categories but *engine_wiring* have a resolution of 2590 × 1942 pixels. For all categories but *pipe_clip* and *pipe_staple*, images were resized to 1000 × 750 pixels. The categories *pipe_clip* and *pipe_staple* show very small defects compared to the scale of the surrounding environment. In order to make sure that the defects can be identified, we used a template matching algorithm to automatically retrieve the inspection area at a size of 400 × 400 pixels and thus remove the rest of the scene. Images for the *engine_wiring* category have an original size of 640 × 480 pixels and were cropped to a size of 400 × 400 pixels.

The test images show a variety of defects depending on the category of items. Some defects represent a missing part or a misplaced item (*tank_screw*, *underbody_pipes*), while others show defective items (*pipe_clip*). Some images also show operators that block the inspection area (*underbody_pipes*). Finally, some pictures show some clipping defects and item position issues (*engine_wiring*). A thorough description of the classes' characteristics is given in Table 4.

All the images of defects correspond to deliberately built anomalies upstream on the assembly line without any modification of the actual shooting conditions. These anomalies correspond to defects that were recorded in the defect library. This protocol enabled us to collect a large number of different images of defects. Naturally, these defects were corrected after shooting.

3.2. Labeling policy

Although unsupervised methods do not require defective items, building a benchmark like AutoVI requires collecting and labeling defects for evaluation purposes. These defects, available only in the test set, must be correctly categorized and segmented. We consider that some of these defects fall under the *logical defect* denomination, as used by Bergmann et al. (2021). Here, we refer to the defects in the *engine_wiring* category, which represent defects in the global structure of the image (positioning of the wires and the blue hoop). We also have a variety of defects representing absent items in the *pipe_staple*, *tank_screw* and *underbody_screw* categories. As such, the precise identification of the defective area is not as straightforward as for *structural defects*, where anomalous elements can be easily identified from their surroundings. Bergmann et al. (2021) generally segment several areas corresponding to a single logical defect, such as a misplaced cable and its normal position at the same time. The anomaly mask is parameterized by a saturation threshold that sets the minimal size of the defect in the segmented area to avoid wrongly penalizing segmentation methods for not segmenting the entire area.

For the *engine_wiring* category, we chose to identify all possible sources of error: namely, the correct possible locations of the blue hoop as well as its incorrect position. We segment the incorrectly fastened cables over a fixed square area alongside the inspection area, so that defective cables are all segmented the same way. Items from *pipe_clip* are segmented both on the clip and on the rubber disk, since visually both areas could be understood to characterize the defect. Items from *pipe_staple* and *tank_screw* are segmented over the area where the staple, or screw, is missing. The segmentation area corresponds to a slightly larger area around the possible positions of the item. Items from *underbody_pipes* are segmented under the line corresponding to the topside of the white platform. Items from *underbody_screw* are segmented over the exact location of the missing screw, using a slightly larger segmentation area.

For missing items (such as the missing screws), the saturation threshold is set to .7 in order to account for the segmentation mask being slightly larger than the smallest item of the defect-free images. For *engine_wiring*, the *blue_hoop* might be misplaced: as such, we have segmented the area where is should be present, and the area where it is present, and set the threshold to the minimum size of a blue hoop as seen in the defect-free images. For *pipe_clip*, the threshold is set to half of the pixels covering the defect area, as the defect can be successfully segmented on one of the two segmented components.

Table 4
Overview of the classes' characteristics.

Class	Environmental variations	Defects encountered
Wiring	Fixed scene. Slight variations in lighting and orientation of the fastening module.	Logical defects. Unfastened cable. Incorrectly placed blue hoop. Apparition of contaminants.
Clip	Slight variations in the position of the inspected item. Variations in the engine model (several modes of normality). Slight differences in lighting.	Small misplaced clip in the pipe connector. Apparition of operators in front of the scene.
Staple	Slight variations in the position of the inspected item. Variations in the engine model (several models of normality).	Missing large staple on the pipe connector.
T. screw	Fixed scene. Variations in the tank model shown (several modes of normality). Moving items in the scene (red receptacle). Apparition of items in certain images at different places (wirings; cables; presence of nuts and bolts). The items never appear in new places in the test set. Slight differences in the exact positioning of the scene. Blurriness present in some pictures.	Absence of the relatively small screw, always in the same location in the image.
Cables	Fixed scene. Largely stable item locations. Apparition of items in certain images (shards). Minor lighting variations.	Presence of the cables in front of the scene. Presence of operators in front of the scene.
U. screw	Fixed scene. Major variations in lighting. Variations in the casing model (several modes of normality). Visible lens flare in some images.	Absent left screw.

Table 5
Example confusion matrix. TP stand for True Positive, FP for False Positive, FN for False Negative and TN for True Negative. Positives stands for defective, and negative for non-defective.

		Ground truth	
		Positive	Negative
Prediction	Positive	TP	FP
	Negative	FN	TN

4. Benchmark study

In this section, we propose a thorough benchmark study over all classes of our dataset to prove its value for overcoming current research challenges.

4.1. Choice of methods

We have chosen six methods that are representative of the different architecture designs reviewed in Section 2.2, and give out the highest AUROCs on MVTec AD and MVTec LOCO as described in Section 2.3: CFlow, DRAEM, DSR, EfficientAD, PaDiM, and Patchcore. The implementation of all methods is taken from the Anomalib library (Akcay et al., 2022). At the time of test, CS-Flow being unavailable, we have chosen CFlow, whose performance is close to CS-Flow. Additionally, we use the EfficientAD-M model, which achieves better performance in terms of AUROC on MVTec AD and MVTec LOCO, at the cost of slightly longer evaluation time.

The methods were first tested on the MVTec dataset to ensure that they achieve their expected performance.

4.2. Choice of metrics

We intend to propose metrics that replicate as closely as possible the direct application of the methods to the industrial context. Concerning image-level detection, with positives representing defective items and negatives non-defective items, the *False Positive Rate* and *False Negative Rate* (FPR, FNR) represent the amount of non-defective items being flagged as defective and defective items being flagged as non-defective, respectively. The two metrics can be directly translated to the associated events on the assembly line, FPR represents the non-defective items being flagged as defective, and FNR represents the defective items that have not been flagged by the inspection system.

Using the definitions given in Table 5, we use the following equations to calculate the FPR and FNR:

$$\mathrm{FPR} = \frac{FP}{FP + TN} \tag{1}$$

$$\mathrm{FNR} = \frac{FN}{FN + TP} \tag{2}$$

As we are advocating for the unsupervised setting, there are no defective items in the training set. In order to obtain the FPR and FNR values, a threshold must be chosen. This threshold may be chosen by maximizing the F-score over the test set for image-level (Božič et al., 2021) and pixel-level metrics (Gudovskiy et al., 2022). Using only the training set, we can derive specific values of FPR by setting manual threshold values. These values can then be used to derive test results without any prior knowledge of the test data. We have chosen the FPR values of 0.01, 0.05, 0.10 (corresponding to True Negative Rates (TNR) of 0.99, 0.95, 0.90).

Additionally, we also use the Area Under the Receiver Operating Curve (AUROC) to compare performances between methods and between datasets as a threshold-agnostic metric. We also use the Average Precision (AP), otherwise known as the Area Under the Precision-Recall Curve (AUPR). The use of both metrics is justified by the fact that we propose a dataset that features high class imbalance between defective and non-defective populations, whose performance can be more accurately assessed by AP, which equally considers the false positive and false negative rates. However, our datasets (as well as all datasets available in the literature) do not show a realistic proportion of defects compared to that which is found in real-world industrial applications. In this sense, AUROC better estimates the performance that would be expected in a real-world system with a different defect prior probability.

Concerning pixel-level detection, a commonly used metric is the Area Under the Per-Region Overlap Curve (AUPRO) (Batzner et al., 2024; Bergmann et al., 2019; Gudovskiy et al., 2022). This metric relies on the Per-Region Overlap (PRO) metric: for m sets of connected pixels marked as anomalous $(A_i), i \in [1, m]$ in a ground truth image, and a set P of pixels predicted as anomalous, the PRO is defined as

$$\mathrm{PRO}(P) = \frac{1}{m} \sum_{i=1}^{m} \frac{|P \cap A_i|}{|A_i|} \ . \tag{3}$$

This measure resembles the TPR at the pixel level, but weighs identically all connected components, regardless of their size. The PRO is the basis for the Saturated Per-Region Overlap (sPRO) measure (Bergmann et al., 2022). Given a set of saturation thresholds $(s_i), i \in [1, m]$, the sPRO is computed as:

$$\mathrm{sPRO}(P) = \frac{1}{m} \sum_{i=1}^{m} \min\left(1, \frac{|P \cap A_i|}{s_i}\right) \ . \tag{4}$$

This measure is used to consider as correctly segmented a zone of which only a proportion $s_i/|A_i|$ is detected. This is particularly relevant for logical defects, for example misplaced items on an image.

Table 6
Classification results measured by AUROCs (in %). The AUROCs shown here are calculated as the means and standard deviations of the AUROCs observed in the eight experiments. Best results are outlined in **bold**. Best results differ from the others according to the Welch's *t*-test at the 5% significance level. If several good results are not significantly different, they are all considered to be the best. The mean AUROCs from the MVTec AD, MVTec LOCO and VisA datasets are reproduced from Table 2.

	CFlow	DRAEM	DSR	Eff.AD	PaDiM	Patchcore
Wiring	46.1±8.6	71.8±2.3	75.9±2.4	**79.7±0.8**	**79.2±1.3**	77.2±0.3
Clip	49.3±12.7	**76.6±3.6**	**81.1±6.8**	**76.8±1.2**	62.6±2.0	73.3±0.9
Staple	51.2±6.7	74.2±12.1	**96.2±3.1**	84.3±1.5	54.6±3.0	92.3±0.6
T. screw	48.7±14.1	63.9±11.8	70.2±10.2	62.4±2.7	49.4±2.7	**89.3±0.7**
Pipes	64.2±19.7	89.2±5.5	56.9±8.6	91.2±1.2	98.9±0.6	**99.8±0.0**
U. screw	48.8±13.0	96.0±2.8	**97.9±2.2**	91.0±0.2	82.6±1.7	**98.9±0.1**
Mean	51.4±5.9	78.6±10.8	79.7±14.3	80.9±9.8	71.2±17.3	**88.4±10.1**
MVTec AD	98.2	98.0	98.2	99.1	97.9	99.0
MVTec LOCO	–	73.6	82.6	90.7	–	80.3
VisA	91.5	–	–	98.1	–	–

Table 7
Classification results measured by AP (in %). The APs shown here are calculated as the means and standard deviations of the APs observed in the eight experiments. Best results are outlined in **bold**. Best results differ from the others according to the Welch's *t*-test at the 5% significance level. If several good results are not significantly different, they are all considered to be the best.

	CFlow	DRAEM	DSR	Eff.AD	PaDiM	Patchcore
Wiring	52.8±6.3	70.2±4.9	**74.2±3.1**	**77.9±1.0**	**77.2±1.3**	**77.5±0.4**
Clip	44.7±11.3	**68.3±6.1**	**69.3±6.8**	59.7±0.9	48.5±1.6	56.0±0.7
Staple	40.1±6.5	57.2±12.8	**94.2±4.5**	61.5±1.9	39.4±1.2	78.0±1.6
T. screw	25.1±7.4	37.2±14.1	34.8±8.9	26.1±1.3	23.0±0.6	**57.9±2.2**
Pipes	66.5±16.9	92.7±2.3	56.9±7.0	88.9±1.2	99.1±0.6	**99.9±0.0**
U. screw	7.2±1.0	**61.1±12.1**	**70.0±18.7**	22.5±0.5	13.9±0.9	**67.7±1.7**
Mean	39.4±19.0	64.4±16.6	**66.6±18.0**	56.1±24.8	50.2±29.7	**72.8±14.8**

In this case, a saturation threshold can be set to qualify the number of pixels that must be detected to consider the region correctly segmented. In Bergmann et al. (2022), the sPRO measure is used to calculate the Area Under the sPRO Curve (AUsPRO), in a similar fashion to AUPRO or AUROC. Given the fact that we also define logical defects with corresponding saturation thresholds and that they are segmented similarly to MVTec LOCO, we will use AUsPRO to estimate the methods' performance for anomaly segmentation.

4.3. Benchmark procedure

We ran eight experiments for each training configuration, evaluating six methods over six classes. The total number of experiments is $8 \times 6 \times 6 = 288$. We used the hyperparameters suggested in the corresponding original papers (Batzner et al., 2024; Defard et al., 2021; Gudovskiy et al., 2022; Roth et al., 2022; Zavrtanik et al., 2021a, 2022). The number of training epochs was set in such a way as to ensure that no model underfits on our dataset. CFlow was trained for 100 epochs, DRAEM for 700 epochs, DSR for 500 epochs, and EfficientAD for 250. PaDiM and Patchcore only require a single training epoch.

All methods were trained and evaluated with an Nvidia V100 GPU and 32Go of RAM.

5. Results

The results displayed in Tables 6 to 9 and Fig. 3 show that existing methods do not attain acceptable results on all the classes of our dataset. We give detailed results for each method below.

5.1. CFlow

CFlow (Gudovskiy et al., 2022) shows the lowest average results on our dataset. The only category for which the AUROC is significantly better than random (0.5) is *underbody_pipes*. CFlow uses multi-scale encodings that are decoded in a normalizing flow, and uses log likelihoods applied to the estimated distribution for multiple scales. CFlow struggles to correctly encode the information, especially at the largest scale, as its activations tend to be extremely susceptible to the variations described in Table 4. As can be seen in Fig. 3, CFlow correctly picks up the absent staple in the *pipe_staple* class, showing that the method is able to identify the anomaly, but is otherwise very susceptible to the variations due to the inability of learning the distribution of an overly irregular population. *underbody_pipes* is the category that show the least variability and shows the best results for CFlow, as it is able to identify the extra cables in most cases.

Furthermore, the AUROC for the *underbody_pipes* class reaches a maximum value of 87.8 and a minimum value of 33.1. These results suggest that, with the right initial weights, CFlow is able to perform on par with methods such as DSR or EfficientAD: however, our results suggest that this method is overly susceptible to initialization conditions.

5.2. DRAEM

DRAEM (Zavrtanik et al., 2021a) shows insufficient performance on most tasks. While DRAEM's AUROCs show that this method is significantly better than average, it does not show satisfactory results in terms of AUROC, AP or TPR.

The autoencoder-based architecture of DRAEM is able to produce very precise heatmaps. However, its sensibility to the variations (see Fig. 3) makes the method prone to false positive activations. In most classes, the defect is identified (missing screws in *tank_screw* and *underbody_screw*, extra pipes in *underbody_pipes*, misaligned staple in *pipe_staple*) but the surrounding noise is overly difficult to represent in the latent space of the autoencoder. DRAEM is completely unable to identify defects such as the misplaced blue hoop or incorrectly fastened cable in *engine_wiring* because it does not leverage enough large-scale spatial information to identify such logical defects.

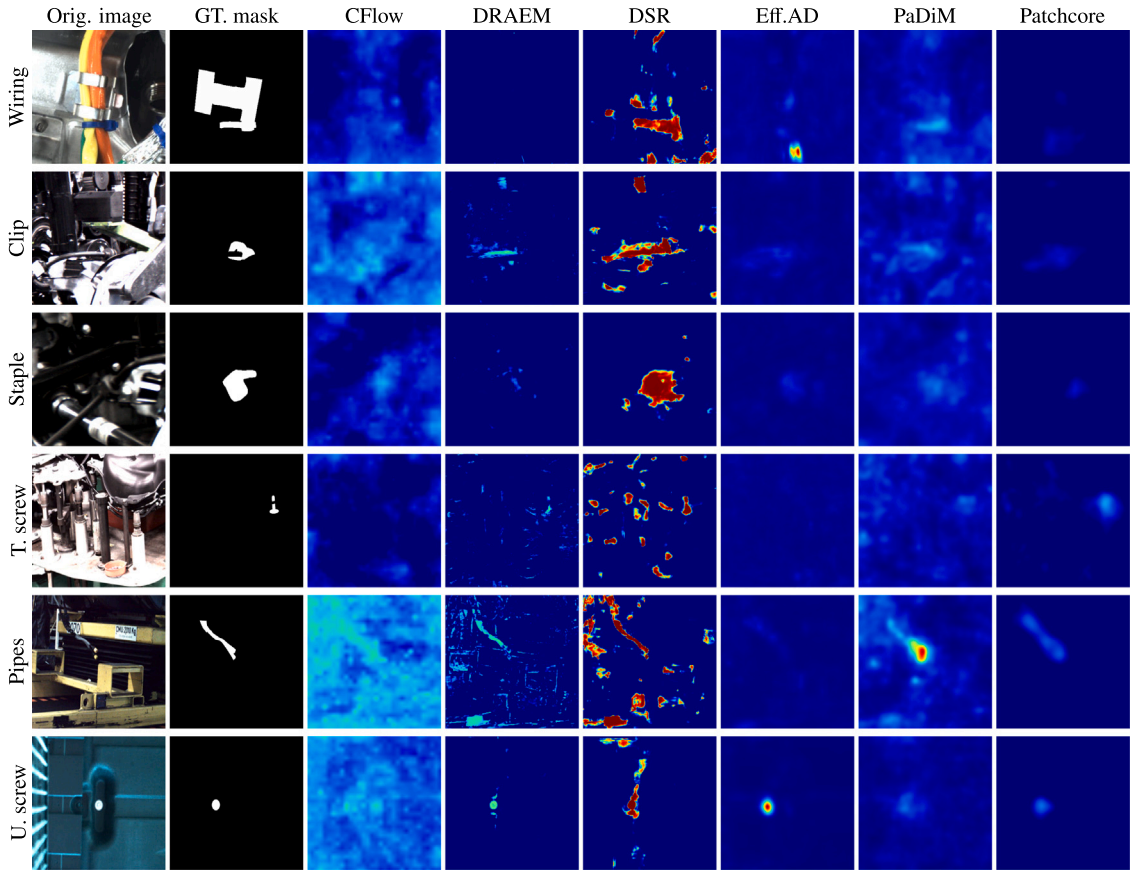


Fig. 3. Example outputs of all tested methods for the categories of the AutoVI dataset, along with the ground truth mask. From left to right: original image reshaped to 256 × 256, ground truth segmentation mask, output heatmaps of CFlow, DRAEM, DSR, EfficientAD, PaDiM and Patchcore.

5.3. DSR

DSR (Zavrtanik et al., 2022) is one of the best-performing methods of our benchmark. DSR consistently ranks among the best methods in terms of AUROC and AP for the *pipe_clip*, *pipe_staple* and *underbody_screw* categories. Namely, DSR reaches a mean TPR of 82.7% at a TNR of 95% for the *pipe_staple* category, largely outperforming the other benchmarked methods. Despite these results, DSR does not achieve good results in terms of TPR at a TNR of 99%, where DSR achieves a mean of at most 63.9% for *underbody_screw*. Besides, DSR is seemingly unable to correctly analyze the *underbody_pipes* category, reaching the lowest mean AUROC out of all tested methods on this category. DSR gives among the best segmentation results of all methods in *pipe_clip*, *pipe_staple* and *tank_screw*.

DSR's representation space works remarkably well to build a reasonable description of the scene. This is particularly visible for the *pipe_staple* class where the absence of the staple exhibits a large response. However, this method shows extreme limitations regarding the identification of the extra pipes in *underbody_pipes*, where it is unable to pick out the correctly-segmented defect from the other responses. DSR struggles with large and complex scenes and relatively small defects (small missing screws in *tank_screw*, thin pipes in *underbody_pipes*), where the representations of the defect, while correct, are drowned out in the surrounding responses. The representation space struggles with reconstructing complex assemblies and identifying the defect within, leading to heatmaps with multiple small pulses scattered throughout the scene.

5.4. EfficientAD

EfficientAD (Batzner et al., 2024) reaches among the highest AUROCs for *pipe_clip*, along with DRAEM and DSR, and *engine_wiring*, along with PaDiM. EfficientAD does not give satisfactory results at high TNR thresholds, not beating 10% TPR at 99% TNR for any category.

EfficientAD is surprisingly unable to specifically identify logical defects, namely the misplaced blue hoops in *engine_wiring*, compared to other methods that have not specifically been designed to identify logical defects, namely PaDiM. Similarly to other methods, EfficientAD shows a sensitivity to the changing environmental conditions in the output heatmaps. These heatmaps show that EfficientAD identifies the defect but the response is very weak, and does not stand out in the surrounding noise. As the image resolution is decreased during computation, the autoencoder and student-teacher may be unable to comprehensively model the input variability and thus struggle to identify the defect. Furthermore, EfficientAD uses a normalization step for the output responses that uses a validation defined as 5% of the training data. The normalization is too spread out due to the variability found in the data: for all defect types, we can see a very small response at the actual defect locations that is drown out in the overly normalized heatmap. We show that normalizing on the entire training dataset increases the performance of EfficientAD on AutoVI in Section 6.4.

5.5. PaDiM

PaDiM (Defard et al., 2021) is the second best-performing method on the *engine_wiring* category, showing some of the most difficult inspection tasks of our dataset, reaching a mean AUROC of 79.2%. Additionally, the variances are very low for all categories in terms of AUROC, AP and TPR, the exception being the *underbody_pipes* and *underbody_screw* categories. However, except for the *underbody_pipes* class, PaDiM does not show especially good results, with the *tank_screw* category having an AUROC at 50%.

Table 8
Classification results measured by True Positive Rates (TPR, in %) for several True Negative Rates (TNR, in %). Best results, outlined in **bold**, differ from the others according to the Welch's t-test at the 5% significance level. If several good results are not significantly different, they are all considered to be best.

	TNR	CFlow	DRAEM	DSR	Eff.AD	PaDiM	Patchcore
Wiring	99	3.0±2.9	4.3±3.3	5.5±1.6	2.9±1.2	5.6±1.0	**8.0±0.6**
	95	6.4±3.0	17.8±3.3	18.5±6.3	**23.1±2.6**	19.8±2.7	**25.5±1.7**
	90	10.9±4.6	27.9±3.9	28.3±8.0	**43.5±3.6**	36.7±5.6	34.9±1.2
Clip	99	**2.6±4.7**	**4.1±5.6**	2.4±1.4	1.0±0.3	0.5±0.3	0.7±0.0
	95	8.0±10.1	**25.4±11.2**	**18.3±7.0**	3.9±0.8	5.1±2.0	4.9±0.0
	90	12.0±12.8	**40.9±12.0**	**42.5±9.4**	11.2±2.9	11.3±2.8	13.1±0.8
Staple	99	2.3±3.9	5.1±11.0	**56.2±22.9**	1.7±0.4	1.0±0.8	3.1±0.6
	95	4.3±4.5	14.6±14.7	**82.7±12.2**	5.3±0.8	4.0±1.5	19.2±5.2
	90	9.9±6.2	31.5±19.4	**89.5±8.0**	9.6±1.1	8.5±1.0	70.6±7.9
T. screw	99	**1.7±3.2**	**4.2±7.1**	1.0±0.0	0.0±0.0	1.2±0.3	**3.4±1.7**
	95	4.2±5.8	9.8±12.8	6.2±7.7	2.2±0.6	4.3±1.8	**20.0±3.0**
	90	9.2±7.1	18.6±18.9	17.2±10.6	7.0±0.7	9.3±1.2	**45.3±4.8**
Pipes	99	8.6±9.5	38.9±15.6	4.1±2.7	8.6±5.1	85.3±11.6	**98.3±0.0**
	95	17.2±15.6	52.4±23.1	6.9±3.9	51.7±6.4	95.0±3.5	**99.5±0.0**
	90	23.2±20.3	66.0±19.9	10.0±6.4	72.7±8.6	97.5±1.1	**99.5±0.0**
U. screw	99	0.0±0.0	**39.6±27.1**	**63.9±32.9**	0.0±0.0	0.0±0.0	16.6±0.0
	95	0.7±1.8	66.0±34.2	88.9±14.4	0.0±0.0	2.1±2.7	**100±0.0**
	90	2.1±2.7	91.7±10.8	97.2±5.6	89.6±10.1	6.2±1.8	**100±0.0**

Table 9
Segmentation results measured by mean AUsPRO (in %), calculated on the eight experiments. The sPRO curves were computed up to the pixel FPR threshold of 5%. Best results are outlined in **bold**. Best results differ from the others according to the Welch's t-test at the 5% significance level. If several good results are not significantly different, they are all considered to be the best.

	CFlow	DRAEM	DSR	Eff.AD	PaDiM	Patchcore
Wiring	5.3±2.0	7.7±1.3	29.4±2.6	**45.5±1.5**	**44.4±3.9**	35.7±0.3
Clip	6.8±5.5	26.4±16.0	**59.6±11.0**	45.9±2.1	38.1±5.6	**60.2±5.7**
Staple	5.9±4.2	55.8±16.1	**78.3±3.9**	49.1±25.6	19.1±5.6	60.3±5.2
T. screw	4.3±16.1	91.1±1.7	**94.2±1.0**	66.3±4.6	50.4±8.1	80.6±0.5
Pipes	53.6±18.4	53.0±5.4	63.1±3.4	55.9±1.5	**78.2±1.7**	**77.6±0.2**
U. screw	59.9±26.2	72.4±2.3	78.4±2.1	98.4±1.2	93.3±1.6	**99.3±0.0**
Mean	22.6±24.2	51.1±27.6	**67.2±20.3**	**60.1±18.5**	53.9±24.8	**68.9±20.0**

PaDiM shows a comparatively good capacity at modeling the Gaussian parameters of the patches of classes that show a stable scene: *engine_wiring*, *underbody_pipes* and *underbody_screw*. Additionally, Defard et al. (2021) suggest that PaDiM is more resilient to rotations of the focal plane than other methods. This may explain the better results of PaDiM on *engine_wiring* than most other classes, as it shows small changes in rotations regarding the positioning of the cables, the metallic clamps and the blue hoop. This method is still unable to consistently detect logical defects. Similarly, PaDiM shows good capacity at identifying the defects in the *underbody_pipes* class, due to their relative simplicity in terms of positioning, stability and environmental variations. Other classes show too much diversity for PaDiM to accurately estimate the Gaussian parameters.

5.6. Patchcore

Patchcore (Roth et al., 2022) is the overall best-performing method, ranking among the best methods in terms of AUROC for *tank_screw*, *underbody_pipes* and *underbody_screw* and reaching high TPRs for *engine_wiring* and *pipe_staple*. Most notably, Patchcore reaches a TPR of 98.3% on *underbody_pipes* for a TNR of 99%. Patchcore is the only method in our benchmark that reaches a TPR of more than 99% for a TNR of 95%, and does so for two classes. Additionally, segmentation results show that Patchcore is able to consistently identify the anomaly, albeit with an imprecise boundary because of its patch-based segmentation algorithm. Patchcore is less adapted to the *pipe_clip* task, where it is not able to accurately identify the defect compared to DRAEM, DSR and EfficientAD in terms of AUROC and AP.

Patchcore uses patch representations: it shows consistently good results as its patch representations enable the method to find defects at fixed locations. Due to its local patch-based architecture, it does not pick up the *engine_wiring* logical defects, which require a way to handle global structures. The misaligned clips in the *pipe_clip* category are identified to some extent by Patchcore, but are too small and difficult to pick out from their surroundings. Patchcore does not issue sufficient distance from the defective patch to its memory bank patches to identify this kind of defects, because it is very similar to correct visual structures seen during training. Patchcore would require a much more precise model of the nominal class patches as seen in the training data to be able to achieve better results.

5.7. Training and evaluation times

Table 10 shows the mean training and evaluation times of all tested methods on our dataset. At evaluation time, all methods are able to process at least several images per second, meaning that all methods are able to be used on high-cadence assembly lines. For all methods, mean training times stand under the ten-hour mark, which is reasonably low for implementing a new inspection cell. Namely, Patchcore and PaDiM take under one minute to train, which means that these methods could be deployed extremely rapidly on a new assembly line.

6. Further experiments

In order to bolster our benchmark study, we present results gathered by varying the inspection and training conditions, in order to identify possible approaches to improve the performance of unsupervised methods on real-world data.

Table 10
Average training time (in seconds) and average number of images processed per second at evaluation time for each method.

	CFlow	DRAEM	DSR	Eff.AD	PaDiM	Patchcore
Training time (in s)	19,374±5,551	29,438±7,357	16,839±4,227	15,954±2,303	15±2	46±23

Table 11
Classification results measured by AUROCs (in %), training with data augmentation. The AUROCs are given as the area under the averaged ROC curve over the eight experiments. Best results are outlined in **bold**. Results that improved with data augmentation are underlined.

	CFlow	DRAEM	DSR	Eff.AD	PaDiM	Patchcore
Wiring	46.6±6.1	70.6±2.4	65.4±6.8	**79.2±0.9**	75.9±1.2	76.5±0.6
Clip	54.6±14.3	**75.5±6.0**	70.6±7.4	**77.8±1.2**	62.3±3.2	68.4±0.8
Staple	53.5±8.5	**98.1±1.7**	85.5±11.3	94.6±0.7	56.4±3.3	92.1±1.0
T. screw	45.7±7.9	64.8±6.1	64.7±10.3	50.2±3.0	46.9±0.9	**86.6±1.3**
Pipes	82.7±7.3	86.8±4.6	56.9±7.2	88.1±2.9	97.9±0.8	**99.8±0.0**
U. screw	60.3±18.5	**98.7±1.0**	**97.4±6.6**	91.3±0.3	79.4±2.0	**98.2±0.6**

6.1. Data augmentation

Data augmentation refers to techniques used to increase the number of available training images. Data augmentation is widely used in industrial applications to counter the problem of imbalanced data or insufficient training data (Kim et al., 2023; Niu et al., 2023).

Bergmann et al. (2022) propose a data augmentation pipeline for benchmarking MVTec LOCO based on vertical and horizontal flips, random rotations and color jitters. We have decided not to include flips and color jitters, as our data is not axially symmetric, and is partially dependent on color cues for anomaly detection. Instead, we have opted for Gaussian noise to simulate the loss of quality that could be occasionally experienced in an industrial environment. Our data augmentation pipeline was set as follows:

- with probability 0.5, a random rotation following a uniform distribution on the interval $[+15°, -15°]$,
- with probability 0.5, a Gaussian noise with individual pixel perturbations p following a Gaussian distribution $p \sim \mathcal{N}(0, \sigma^2)$ with $\sigma^2 \sim U(10, 120)$, σ^2 being sampled once per image.

The two processes are independent, so both augmentations can be applied on the same image.

Table 11 summarizes the classification results with data augmentation. Results show that using data augmentation leads to overall worse classification results in terms of mean AUROC. Some categories (*engine_wiring*, *pipe_staple*) show a significant rotation variability between shots. Only CFlow, DRAEM and, to some extent, EfficientAD benefit from data augmentation. These methods seem to benefit from extra diversity in the training data; CFlow might be able to better estimate the data distribution, while DRAEM is able to construct a better representation in its latent space. EfficientAD is able to largely improve its results on *pipe_staple*, benefiting from the extra rotation. However, its performance on *tank_screw* decreases sharply, suggesting that learning using rotated data does not help the method locate the missing screw. On the other hand, DSR, an autoencoder-based method, like DRAEM, shows a significant decrease in performance: this might be due to the fact that DSR uses quantized subspace representations (Razavi et al., 2019) which might be overly sensitive to the Gaussian noise. Finally, PaDiM and Patchcore do not show significant difference with data augmentation. PaDiM arguably does not benefit from extra rotation information, due to the purported resistance of this method to rotation variations (Defard et al., 2021). Finally, the issue of Patchcore is mostly linked to logical defect detection and small defect identification: the augmentation that was carried out does not answer the limitations of the method.

6.2. Varying the window size

The categories *pipe_clip* and *pipe_staple* show images that were cropped from a much larger image (see Fig. 4), as well as *engine_wiring* which is cropped from a comparatively smaller image. We have tested the performance of the algorithms using different window sizes around the defect, in order to identify the scale at which the tested algorithms react to the presence of the defect.

For *pipe_clip* and *pipe_staple*, we have tested crops at resolutions of 200 × 200, 400 × 400 and 600 × 600 pixels. For *engine_wiring*, we have tested the original, uncropped image of size 640 × 480 pixels, and crops of sizes 400 × 400 and 400 × 480 pixels. The other images have not been resized due to the defect being large enough to be picked up. We have trained and evaluated each method eight times on each category and each size.

Table 12 shows the results of our experiments for different window sizes. The influence of the window size is minor for the *engine_wiring* category, where the best-performing methods for the category, EfficientAD and PaDiM, keep stable AUROCs throughout the experiments. However, other methods see their AUROC decrease as the window size is increased. For the *pipe_clip* category, most methods see a steady decrease in AUROC as the window size increases, except for DSR which achieves its best results for a window size of 400 × 400, although with a high standard deviation. Finally, the *pipe_staple* category also sees a significant decrease in its AUROC as the window size increases, although DRAEM performs significantly better than other methods for a window size of 600 × 600.

For the *engine_wiring* class, differences are generally minor due to the small differences in size. However, these results show that all methods are resilient to the differences that are removed in the cropping operation (apparition of nuts and bolts, shadows and lighting differences, positioning of the wires). For the *pipe_clip* class, results decrease as expected due to the increasing complexity of the image and relative difficulty of this specific detection task. Finally, the *pipe_staple* class shows an outlier result in DRAEM's performance at the largest tested cropping size, while the better performing DSR falls short. We have established that DSR fails to perform with complex scenes and relatively small defects, while DRAEM seems to perform better on these tasks while struggling with location information. At a large cropping size, thus, DSR falls short due to being confronted to a much larger scene and smaller defect, while DRAEM gives an essentially stable result.

6.3. Number of training images

We have experimented with reducing the number of training images. This approach, dubbed *few-shot anomaly detection*, is of high interest for industrial applications as it would allow defect detection

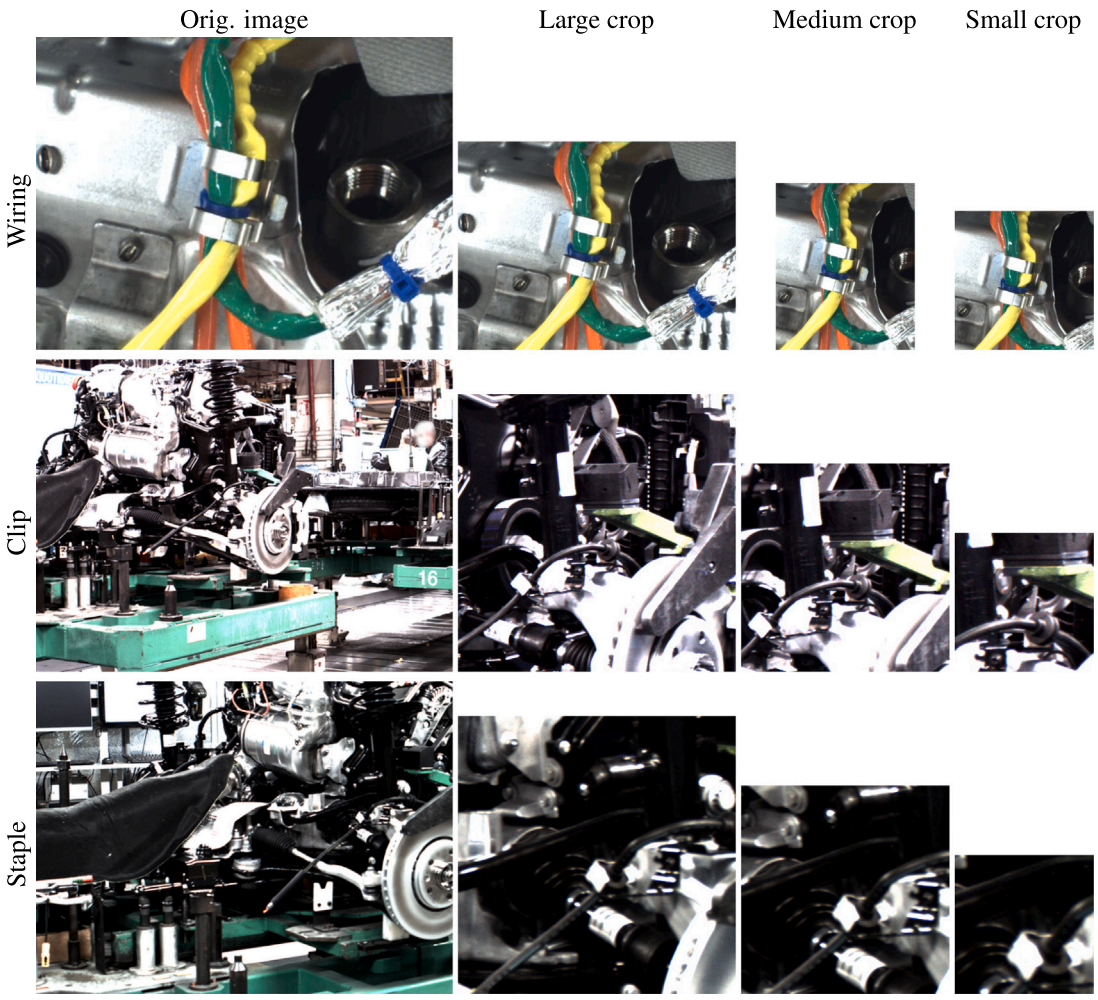


Fig. 4. Comparison of images before and after cropping.

Table 12
Classification results measured by AUROCs (in %) for different window sizes. The AUROCs shown here are calculated as the means and standard deviations of the AUROCs observed in the eight experiments. Best results are outlined in **bold**. Best results differ from the others according to the Welch's *t*-test at the 5% significance level. If several good results are not significantly different, they are all considered to be the best.

	CFlow	DRAEM	DSR	Eff.AD	PaDiM	Patchcore
Wiring 400 × 400	46.1±8.6	71.8±2.3	75.9±2.4	**79.7±0.8**	**79.2±1.3**	77.2±0.3
Wiring 400 × 480	51.4±8.5	69.4±1.1	75.5±1.7	**80.7±0.6**	79.8±0.4	77.7±0.4
Wiring 640 × 480	37.8±10.2	68.4±3.0	72.1±2.9	**80.6±0.7**	**80.0±1.2**	76.3±0.4
Clip 200 × 200	61.8±8.7	76.3±6.8	76.1±3.3	**83.5±1.0**	70.5±2.9	79.5±0.4
Clip 400 × 400	49.3±12.7	**76.6±3.6**	**81.1±6.8**	**76.8±1.2**	62.6±2.0	73.3±0.9
Clip 600 × 600	50.9±6.0	**66.6±9.8**	**71.5±3.9**	58.8±1.0	53.5±2.1	58.2±0.9
Staple 200 × 200	48.7±7.8	88.1±6.7	**98.7±2.9**	82.6±2.4	57.5±2.0	**97.0±0.3**
Staple 400 × 400	51.2±6.7	74.2±12.1	**96.2±3.1**	84.3±1.5	54.6±3.0	92.3±0.6
Staple 600 × 600	37.4±5.6	**82.7±12.8**	60.9±9.1	73.5±5.4	30.0±2.0	52.9±1.4

using as few training images as possible (Liu et al., 2023). We ran eight experiments using randomly selected training images on each category of AutoVI using the best-performing method for classification, Patchcore.

Fig. 5 shows the results of our experiments. In the majority of cases, a few-shot configuration with less than 10 training images leads to significantly worse results than using the whole training dataset. The only exception is the *underbody_pipes* class, with an AUROC above 99% with only one training image. In all other cases, the AUROC increases steadily with the number of training images.

For the *engine_wiring*, *pipe_clip*, *pipe_staple* and *tank_screw* categories, the images show significant variations and/or exhibit difficult anomalies, which explains the poor performance of few-shot training configurations. The results can reach an AUROC as low as 0.5, which means that the classifier does not perform better than a random classifier and does not learn enough information to discern defects from nominal patches. For the *tank_screw* category, the AUROCs are even significantly below 0.5 with 4 or fewer training images. Due to the variability offered in these categories, Patchcore identifies a large number of false positives that correspond to normal variations to such an extent that

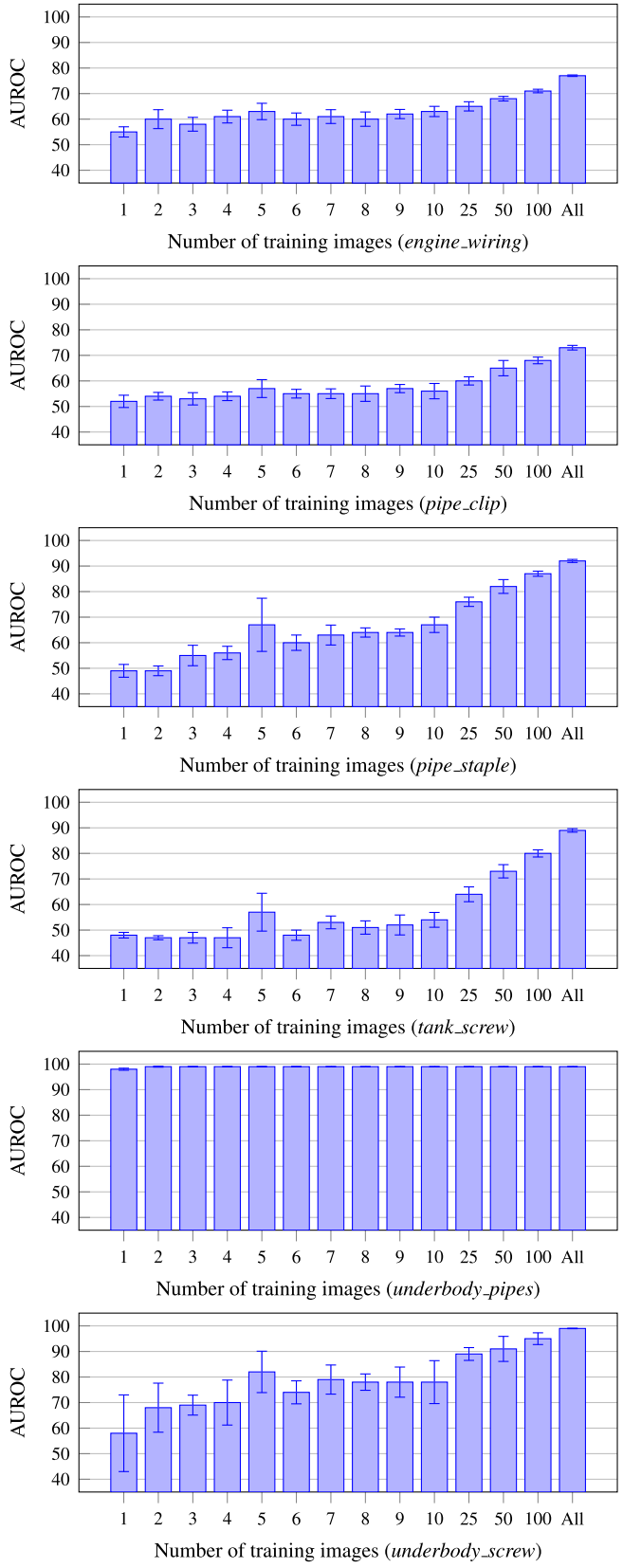


Fig. 5. Performance of the Patchcore algorithm for different few-shot learning configurations.

Table 13
Classification results measured by AUROCs (in %) for EfficientAD, normalized on the entire training set. The AUROCs shown here are calculated as the means and standard deviations of the AUROCs observed in the eight experiments. Best results are outlined in **bold**.

	Eff.AD, extra norm.	Eff.AD, no extra norm.
Wiring	**80.6±0.6**	79.7±0.8
Clip	**79.5±1.5**	76.8±1.2
Staple	**87.0±0.7**	84.3±1.5
T. screw	**63.8±2.5**	**62.4±2.7**
Pipes	**88.4±4.3**	**91.2±1.2**
U. screw	**90.5±0.4**	**91.0±0.2**

it falls short of the random classifier. For the *underbody_pipes* category, the excellent performance of the 1-shot configuration can be explained by the fact that the defects shown are structural defects that take up a significant portion of the image, thus making them stand out in comparison to the rest of the scene, in addition to the scene itself not showing significant variability. Finally, the *underbody_screw* shows a scenario that attains excellent results for classification with the full training dataset, but very low results in few-shot configurations. This category exhibits very significant variations: colors changing over time, presence of lens flare in some images, etc. Although the defect is easy to identify, it is difficult or even impossible for a detection algorithm to handle such a variety of correct conditions with only a few training images. The high variability suggests that depending on the selected training images, the algorithm will correctly identify defects only in images similar to the training images; as there is a different number of pictures in different lighting configurations, the AUROC will change significantly between runs.

6.4. Additional normalization data for EfficientAD

We show in Table 13 that EfficientAD yields better results when normalizing the branches' responses on the entire training dataset, instead of the validation set consisting of 5% of the training data suggested by the authors (Batzner et al., 2024). Namely, we see that EfficientAD achieves the best performance out of all methods on the *engine_wiring* category. This corresponds to the fact that the normalization step requires more data to account for the extra variability found in our dataset. As such, the only classes for which the normalization does not improve results are *underbody_pipes* and *underbody_screw*, which show relatively little structural variability.

7. Discussion and conclusion

We have introduced the Automotive Visual Inspection Dataset (AutoVI), a genuine real-world industrial production dataset for visual inspection. Our dataset includes six different inspection tasks gathered from real assembly lines, including realistic defective parts that were directly created on the assembly line. To the best of our knowledge, this is the first publicly available dataset that shows a variety of in-situ industrial tasks, including large assemblies, part checks, and inclusion of logical defects.

Our benchmark study shows several interesting results concerning the performance of existing algorithms on real-world tasks with minimal tuning of both the data and the methods. We see that some methods achieve excellent detection results on some tasks, namely Patchcore for the *underbody_screw* class, with a True Positive Rate of over 98% at a True Negative Rate of 99%. However, all other tasks do not show such results, and leave considerable room for improvement.

Further experiments have comforted our analysis of the tested methods. We show that all methods are sensitive to different challenges that stem from a real-world, industrial case, notably the relative complexity of the scenes, the size of the defect, and the large-scale positioning to

Table 14
Suggested usage of the reviewed algorithms based on the results of our benchmark study.

	CFlow	DRAEM	DSR	Eff.AD	PaDiM	Patchcore
High variability			✓			✓
Small defects		✓		✓	✓	✓
Complex scenes			✓	✓		✓
Struct. defects	✓	✓	✓	✓	✓	✓
Logical defects				✓		

detect logical defects. Clearly, there is currently no method that fits all types of defect detection problem, Table 14 shows the methods for obtaining satisfactory results depending on the characteristics of the problem to be solved. Overall, Patchcore is able to work well in large and complex scenes (*tank_screw*, *underbody_pipes*), as well as relatively straightforward detection tasks (large staple in *pipe_staple*, large screw in *underbody_screw*). Detection tasks such as small, locally difficult defects in complex scenes (*pipe_clip*) or logical defects featuring wrongly connected cables and hoops (*engine_wiring*) are more difficult for this method. Autoencoders such as DRAEM, DSR and in part EfficientAD are able to better identify the local defective structure in *pipe_clip*, while DSR is particularly apt at detecting the missing staple in *pipe_staple*: though it constitutes a relatively large defect, the spatial structure of DSR's subspaces allow it to specifically focus on all parts of the image and pick up these defects. Finally, PaDiM shows slightly better performance than other methods on *engine_wiring* due to its relative stability to rotation that is visible on some images. Most other methods show results that fall short of these three method's performances for all benchmarked tasks. Note also that the line headers in Table 14 can also be used as features of defect detection problems. These features are useful for comparing method performance across benchmarks: if the current benchmark differs, some of their individual tasks may share certain common properties explaining similar trends across benchmarks.

Finally, for most categories, a consequential amount of training data is still required, as removing images from the training set significantly reduces the detection performance. Training with only a few images, known as *few-shot learning*, would be an interesting perspective as it would allow an inspection system to start detection while having only seen a few images, saving time on the installation of the system.

Performance-wise, all methods have shown a training time of at most ten hours, and an evaluation time well under the second, which makes all tested methods fit for use in a high-cadence industrial context. Further experiments should be conducted on lower-tier hardware to imitate embedded systems' performances that are most likely used in an industrial detection system.

Table 6 shows that the methods tested give results largely inferior results compared to MVTec AD, and results that are comparable to MVTec LOCO (lower for DSR and EfficientAD, higher for DRAEM and Patchcore) despite the lower prevalence of logical defects in AutoVI. In particular, we show that the best AUROC of AutoVI is 88.4% with Patchcore, while the best AUROCs of MVTec AD, MVTec LOCO and VisA are respectively 99.1%, 90.7% and 98.1% with the EfficientAD method. This result illustrates the fact that AutoVI shows a problem that is difficult and different than the ones showcased in existing benchmarks, seeing as the best method on AutoVI is different from the best method on the MVTec and VisA datasets.

AutoVI has allowed us to make a comprehensive benchmark of state-of-the-art unsupervised anomaly detection algorithms on complex, real-world problems. The building of AutoVI has required several months of shooting, selecting, labeling images, and contains a number of images similar to other major datasets in the literature, such as MVTec LOCO (Bergmann et al., 2022). We have proposed the publication of AutoVI as a public benchmark to bolster research efforts towards development of new unsupervised defect detection methods that will be better adapted to real-world tasks.

Glossary

- **Logical defect**: Defect that stems from an erroneous assembly of items in the scene. Associated with large-scale structure identification. See Bergmann et al. (2022).
- **Structural defect**: Defect that stems from a local, visual defect on the scene. Associated with local visual structures. See Bergmann et al. (2022).
- **Unsupervised defect detection**: Machine learning framework where methods learn only from non-defective data.

CRediT authorship contribution statement

Philippe Carvalho: Writing – review & editing, Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Meriem Lafou:** Writing – review & editing, Validation, Supervision, Project administration, Funding acquisition, Data curation, Conceptualization. **Alexandre Durupt:** Writing – review & editing, Validation, Supervision, Resources, Project administration, Funding acquisition, Data curation, Conceptualization. **Antoine Leblanc:** Writing – review & editing, Validation, Supervision, Resources, Project administration, Funding acquisition, Conceptualization. **Yves Grandvalet:** Writing – review & editing, Validation, Methodology, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The dataset can be found at the following address: https://zenodo.org/records/10459003.

The code for reproducing the paper's experiments can be found at the following address: https://github.com/phcarval/autovi-paper-code.

We also release the evaluation code to run new experiments on the AutoVI dataset: https://github.com/phcarval/autovi_evaluation_code.

The training and testing code is built upon the Anomalib library. We use a modified version of the code available at https://github.com/openvinotoolkit/anomalib/, based on the commit `a94481b1`.

The AUsPRO computation code is based on the evaluation code made available by MVTec at https://www.mvtec.com/company/research/datasets/mvtec-loco.

Acknowledgments

This work was carried out with the support of Renault Group and the French Agence Nationale de la Recherche [grant number ANR-20-CE10-0004]. This work was granted access to the HPC/AI resources of IDRIS under the allocation AD011012936R1 made by GENCI.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.compind.2024.104151.

References

Akcay, S., Ameln, D., Vaidya, A., Lakshmanan, B., Ahuja, N., Genc, U., 2022. Anomalib: A deep learning library for anomaly detection. http://dx.doi.org/10.48550/arXiv.2202.08341.

Akcay, S., Atapour-Abarghouei, A., Breckon, T.P., 2019. GANomaly: Semi-supervised anomaly detection via adversarial training. In: Jawahar, C.V., Li, H., Mori, G., Schindler, K. (Eds.), Proceedings of the Asian Conference on Computer Vision. ACCV, 11363 LNCS, Springer International Publishing, Cham, pp. 622–637. http://dx.doi.org/10.1007/978-3-030-20893-6_39.

Batzner, K., Heckler, L., König, R., 2024. EfficientAD: Accurate visual anomaly detection at millisecond-level latencies. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. WACV, pp. 128–138. http://dx.doi.org/10.48550/arXiv.2303.14535.

Bergmann, P., Batzner, K., Fauser, M., Sattlegger, D., Steger, C., 2021. The MVTec anomaly detection dataset: A comprehensive real-world dataset for unsupervised anomaly detection. Int. J. Comput. Vis. 129 (4), 1038–1059. http://dx.doi.org/10.1007/s11263-020-01400-4.

Bergmann, P., Batzner, K., Fauser, M., Sattlegger, D., Steger, C., 2022. Beyond dents and scratches: Logical constraints in unsupervised anomaly detection and localization. Int. J. Comput. Vis. 130 (4), 947–969. http://dx.doi.org/10.1007/S11263-022-01578-9.

Bergmann, P., Fauser, M., Sattlegger, D., Steger, C., 2019. 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, IEEE, pp. 9584–9592. http://dx.doi.org/10.1109/CVPR.2019.00982.

Bergmann, P., Fauser, M., Sattlegger, D., Steger, C., 2020. Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. CVPR, IEEE, pp. 4182–4191. http://dx.doi.org/10.1109/CVPR42600.2020.00424.

Božič, J., Tabernik, D., Skočaj, D., 2021. Mixed supervision for surface-defect detection: From weakly to fully supervised learning. Comput. Ind. 129, http://dx.doi.org/10.1016/j.compind.2021.103459.

Carvalho, P., Durupt, A., Grandvalet, Y., 2022. A survey of machine learning approaches for visual inspection on the DAGM dataset. In: 19th International Conference on Manufacturing Research. ICMR2022, In: Advances in Manufacturing Technology XXXV, IOS Press, Derby, United Kingdom, pp. 255–260. http://dx.doi.org/10.3233/ATDE220600.

Carvalho, P., Durupt, A., Grandvalet, Y., 2023. A review of benchmarks for visual defect detection in the manufacturing industry. In: Gerbino, S., Lanzotti, A., Martorelli, M., Mirálbes Buil, R., Rizzi, C., Roucoules, L. (Eds.), Advances on Mechanics, Design Engineering and Manufacturing IV. Springer International Publishing, Cham, pp. 1527–1538. http://dx.doi.org/10.1007/978-3-031-15928-2_133.

Cohn, R., Holm, E., 2021. Unsupervised machine learning via transfer learning and k-means clustering to classify materials image data. Integr. Mater. Manuf. Innov. 10 (2), 231–244. http://dx.doi.org/10.1007/s40192-021-00205-8.

Defard, T., Setkov, A., Loesch, A., Audigier, R., 2021. PaDiM: A patch distribution modeling framework for anomaly detection and localization. In: Del Bimbo, A., Cucchiara, R., Sclaroff, S., Farinella, G.M., Mei, T., Bertini, M., Escalante, H.J., Vezzani, R. (Eds.), Pattern Recognition. ICPR International Workshops and Challenges. vol. 12664 LNCS, Springer International Publishing, Cham, pp. 475–489. http://dx.doi.org/10.1007/978-3-030-68799-1_35.

Dinh, L., Sohl-Dickstein, J., Bengio, S., 2017. Density estimation using real NVP. In: International Conference on Learning Representations. ICLR, http://dx.doi.org/10.48550/arXiv.1605.08803.

Gao, Y., Li, X., Wang, X.V., Wang, L., Gao, L., 2021. A review on recent advances in vision-based defect recognition towards industrial intelligence. J. Manuf. Syst. http://dx.doi.org/10.1016/J.JMSY.2021.05.008.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press, URL: https://www.deeplearningbook.org/.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2020. Generative adversarial networks. Commun. ACM 63 (11), 139–144. http://dx.doi.org/10.1145/3422622.

Gudovskiy, D., Ishizaka, S., Kozuka, K., 2022. CFLOW-AD: Real-time unsupervised anomaly detection with localization via conditional normalizing flows. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. WACV, pp. 1819–1828. http://dx.doi.org/10.1109/WACV51458.2022.00188.

Guo, H., Ren, L., Fu, J., Wang, Y., Zhang, Z., Lan, C., Wang, H., Hou, X., 2023. Template-guided hierarchical feature restoration for anomaly detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. ICCV, pp. 6447–6458. http://dx.doi.org/10.1109/ICCV51070.2023.00593.

He, K., Zhang, X., Ren, S., Sun, J., 2015. Deep residual learning for image recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. CVPR, IEEE, pp. 770–778. http://dx.doi.org/10.1109/CVPR.2016.90.

Hinton, G., Vinyals, O., Dean, J., 2015. Distilling the knowledge in a neural network. NIPS Deep. Learn. Represent. Learn. Work. http://dx.doi.org/10.48550/arXiv.1503.02531.

Jeong, J., Zou, Y., Kim, T., Zhang, D., Ravichandran, A., Dabeer, O., 2023. WinCLIP: Zero-/few-shot anomaly classification and segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. CVPR, IEEE Computer Society, pp. 19606–19616. http://dx.doi.org/10.1109/CVPR52729.2023.01878.

Kim, Y., Lee, T., Hyun, Y., Coatanea, E., Mika, S., Mo, J., Yoo, Y.J., 2023. Self-supervised representation learning anomaly detection methodology based on boosting algorithms enhanced by data augmentation using StyleGAN for manufacturing imbalanced data. Comput. Ind. 153, 104024. http://dx.doi.org/10.1016/J.COMPIND.2023.104024.

Kobyzev, I., Prince, S.J., Brubaker, M.A., 2021. Normalizing flows: An introduction and review of current methods. IEEE Trans. Pattern Anal. Mach. Intell. 43 (11), 3964–3979. http://dx.doi.org/10.1109/TPAMI.2020.2992934.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60 (6), 84–90. http://dx.doi.org/10.1145/3065386.

Lindemann, B., Maschler, B., Sahlab, N., Weyrich, M., 2021. A survey on anomaly detection for technical systems using LSTM networks. Comput. Ind. 131, 103498. http://dx.doi.org/10.1016/J.COMPIND.2021.103498.

Liu, H., Tian, Y., Li, L., Lu, Y., Feng, J., Xi, F., 2023. Full-cycle data purification strategy for multi-type weld seam classification with few-shot learning. Comput. Ind. 150, 103939. http://dx.doi.org/10.1016/J.COMPIND.2023.103939.

Niu, S., Peng, Y., Li, B., Wang, X., 2023. A transformed-feature-space data augmentation method for defect segmentation. Comput. Ind. 147, 103860. http://dx.doi.org/10.1016/J.COMPIND.2023.103860.

Pang, G., Shen, C., Cao, L., Hengel, A.V.D., 2021. Deep learning for anomaly detection: A review. ACM Comput. Surv. 54 (2), 1–38. http://dx.doi.org/10.1145/3439950.

Papamakarios, G., Nalisnick, E., Rezende, D.J., Mohamed, S., Lakshminarayanan, B., 2021. Normalizing flows for probabilistic modeling and inference. J. Mach. Learn. Res. 22 (1), 2617–2680. http://dx.doi.org/10.5555/3546258.3546315.

Perlin, K., 1985. An image synthesizer. SIGGRAPH Comput. Graph. 19 (3), 287–296. http://dx.doi.org/10.1145/325165.325247.

Rački, D., Tomaževič, D., Skočaj, D., 2018. A compact convolutional neural network for textured surface anomaly detection. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. WACV, IEEE, pp. 1331–1339. http://dx.doi.org/10.1109/WACV.2018.00150.

Razavi, A., van den Oord, A., Vinyals, O., 2019. Generating diverse high-fidelity images with VQ-VAE-2. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché Buc, F., Fox, E., Garnett, R. (Eds.), Advances in Neural Information Processing Systems. vol. 32, Curran Associates, Inc., pp. 14837–14847. http://dx.doi.org/10.48550/arXiv.1906.00446.

Rippel, O., Mertens, P., Merhof, D., 2020. Modeling the distribution of normal data in pre-trained deep features for anomaly detection. In: International Conference on Pattern Recognition. ICPR, IEEE, pp. 6726–6733. http://dx.doi.org/10.1109/ICPR48806.2021.9412109.

Roth, K., Pemula, L., Zepeda, J., Schölkopf, B., Brox, T., Gehler, P., 2022. Towards total recall in industrial anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. CVPR, pp. 14318–14328. http://dx.doi.org/10.1109/CVPR52688.2022.01392.

Rudolph, M., Wehrbein, T., Rosenhahn, B., Wandt, B., 2022. Fully convolutional cross-scale-flows for image-based defect detection. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. WACV, pp. 1088–1097. http://dx.doi.org/10.1109/WACV51458.2022.00189.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L., 2015. ImageNet large scale visual recognition challenge. Int. J. Comput. Vis. 115, 211–252. http://dx.doi.org/10.1007/S11263-015-0816-Y.

Schlegl, T., Seeböck, P., Waldstein, S.M., Langs, G., Schmidt-Erfurth, U., 2019. f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks. Med. Image Anal. 54, 30–44. http://dx.doi.org/10.1016/J.MEDIA.2019.01.010.

Severstal, 2019. Severstal: Steel defect detection. URL: https://www.kaggle.com/c/severstal-steel-defect-detection/overview/description.

Shi, X., Zhang, S., Cheng, M., He, L., Tang, X., Cui, Z., 2023. Few-shot semantic segmentation for industrial defect recognition. Comput. Ind. 148, 103901. http://dx.doi.org/10.1016/J.COMPIND.2023.103901.

Song, K., Yan, Y., 2013. A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects. Appl. Surf. Sci. 285, 858–864. http://dx.doi.org/10.1016/j.apsusc.2013.09.002.

Tabernik, D., Šela, S., Skvarč, J., Skočaj, D., 2020. Segmentation-based deep-learning approach for surface-defect detection. J. Intell. Manuf. 31 (3), 759–776. http://dx.doi.org/10.1007/s10845-019-01476-x.

Wang, T., Chen, Y., Qiao, M., Snoussi, H., 2018. A fast and robust convolutional neural network-based defect detection model in product quality control. Int. J. Adv. Manuf. Technol. 94 (9–12), 3465–3471. http://dx.doi.org/10.1007/s00170-017-0882-0.

Wang, L., Yoon, K.-J., 2022. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. IEEE Trans. Pattern Anal. Mach. Intell. 44 (6), 3048–3068. http://dx.doi.org/10.1109/TPAMI.2021.3055564.

Weimer, D., Scholz-Reiter, B., Shpitalni, M., 2016. Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection. CIRP Ann 65 (1), 417–420. http://dx.doi.org/10.1016/j.cirp.2016.04.072.

Wieler, M., Hahn, T., Hamprecht, F.A., 2007. Weakly supervised learning for industrial optical inspection. In: 29th Annual Symposium of the German Association for Pattern Recognition. http://dx.doi.org/10.5281/zenodo.8086136.

Yao, H., Luo, W., Yu, W., Zhang, X., Qiang, Z., Luo, D., Shi, H., 2023. Dual-attention transformer and discriminative flow for industrial visual anomaly detection. IEEE Trans. Autom. Sci. Eng. http://dx.doi.org/10.1109/TASE.2023.3322156.

Zavrtanik, V., Kristan, M., Skočaj, D., 2021a. DRAEM – a discriminatively trained reconstruction embedding for surface anomaly detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. ICCV, IEEE, pp. 8330–8339. http://dx.doi.org/10.1109/ICCV48922.2021.00822.

Zavrtanik, V., Kristan, M., Skočaj, D., 2021b. Reconstruction by inpainting for visual anomaly detection. Pattern Recognit. 112, http://dx.doi.org/10.1016/j.patcog.2020.107706.

Zavrtanik, V., Kristan, M., Skočaj, D., 2022. DSR – a dual subspace re-projection network for surface anomaly detection. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (Eds.), Proceedings of the European Conference on Computer Vision. ECCV, Springer Nature Switzerland, Cham, pp. 539–554. http://dx.doi.org/10.1007/978-3-031-19821-2_31.

Zeiser, A., Özcan, B., van Stein, B., Bäck, T., 2023. Evaluation of deep unsupervised anomaly detection methods with a data-centric approach for on-line inspection. Comput. Ind. 146, 103852. http://dx.doi.org/10.1016/J.COMPIND.2023.103852.

Zhang, J., Suganuma, M., Okatani, T., 2024. Contextual affinity distillation for image anomaly detection. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. CVPR, pp. 149–158. http://dx.doi.org/10.48550/arXiv.2307.03101.

Zhang, Z., Zhao, Z., Zhang, X., Sun, C., Chen, X., 2023. Industrial anomaly detection with domain shift: A real-world dataset and masked multi-scale reconstruction. Comput. Ind. 151, 103990. http://dx.doi.org/10.1016/J.COMPIND.2023.103990.

Zou, Y., Jeong, J., Pemula, L., Zhang, D., Dabeer, O., 2022. Spot-the-difference self-supervised pre-training for anomaly detection and segmentation. In: Proceedings of the European Conference on Computer Vision. ECCV, Springer, Cham, pp. 392–408. http://dx.doi.org/10.1007/978-3-031-20056-4_23.