Computers & Industrial Engineering 177 (2023) 109045

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Anomaly detection for industrial quality assurance: A comparative evaluation of unsupervised deep learning models

Justus Zipfel [a], Felix Verworner [a], Marco Fischer [a], Uwe Wieland [b], Mathias Kraus [c], Patrick Zschech [c,*]

[a] Software Development Center, Volkswagen AG, Dresden, Germany
[b] Hochschule für Technik und Wirtschaft, Dresden, Germany
[c] Friedrich-Alexander-Universität Erlangen-Nürnberg, Nürnberg, Germany

ARTICLE INFO

Keywords:
Computer vision
Quality control
Industrial inspection
Anomaly detection
Deep learning
Unsupervised machine learning

ABSTRACT

Across many industries, visual quality assurance has transitioned from a manual, labor-intensive, and error-prone task to a fully automated and precise assessment of industrial quality. This transition has been made possible due to advances in machine learning in general, and supervised learning in particular. However, the majority of supervised learning approaches only allow to identify pre-defined categories, such as certain error types on manufactured objects. New, unseen error types are unlikely to be detected by supervised models. As a remedy, this work studies unsupervised models based on deep neural networks which are not limited to a fixed set of categories but can generally assess the overall quality of objects. More specifically, we use a quality inspection case from a European car manufacturer and assess the detection performance of three unsupervised models (i.e., Skip-GANomaly, PaDiM, PatchCore). Based on an in-depth evaluation study, we demonstrate that reliable results can be achieved with fully unsupervised approaches that are even competitive with those of a supervised counterpart.

1. Introduction

Industrial quality assurance plays a fundamental role in production processes as it helps manufacturers to improve product reliability, reduce waste, and increase the overall efficiency of the production environment. Furthermore, it ensures that products meet customer requirements and industry standards, which is crucial to protect the manufacturer's reputation and prevent costly recalls or legal issues (Zonnenshain & Kenett, 2020). In this vein, visual quality inspection has evolved to an important component because the optical properties of a product are directly linked to its quality and the perceived image of the company. Therefore, visual inspections aim at a fast and accurate exploration of visually impaired products in order to detect and eliminate any qualitative defects before the products are sold and delivered (Avola et al., 2022; Olimov et al., 2022).

Traditional inspection methods involve human labor to detect defects and deviations. However, since manual inspections are considered time-consuming, tiring, and error-prone, modern strategies increasingly rely on algorithmic approaches from the field of anomaly detection (Chandola et al., 2009; Pang et al., 2021). Of particular interest are deep neural networks from the discipline of machine learning and deep learning that are able to learn non-trivial relationships from large data collections (Janiesch et al., 2021; Kraus et al., 2020). Thus, by training a deep neural network with known examples of normal and abnormal instances, it can capture fine-grained patterns in high-dimensional image features and create a model that is capable of distinguishing defective from non-defective objects (LeCun et al., 2015). This principle is generally known as *supervised learning*, which has led to remarkable performance results in industrial contexts (e.g., Harl et al., 2022; Lin et al., 2021; Lopes & Alexandre, 2020; Tabernik et al., 2019; Tao et al., 2018).

However, a major drawback of supervised learning methods is that they require labeled examples of damage patterns that are already given during training in order to be able to correctly classify such cases in later operational use (Janiesch et al., 2021; Kraus et al., 2020). In practice, however, labeled instances of relevant damage patterns are rarely available due to a high degree of standardization of industrial processes. That is, deviations from normal conditions occur very rarely, making it nearly impossible to collect a sufficient number of labeled examples that accurately reflect representative types of defects. Similarly, it is a costly and time-consuming endeavor to synthetically generate and label anomalous samples that can be used for model training.

* Correspondence to: Lange Gasse 20, 90403 Nürnberg, Germany.
E-mail address: patrick.zschech@fau.de (P. Zschech).

https://doi.org/10.1016/j.cie.2023.109045
Received 24 August 2022; Received in revised form 20 December 2022; Accepted 24 January 2023
Available online 31 January 2023

Besides, it is often not even known in advance which types of defects may develop in the course of dynamic production settings, so that supervised learning methods quickly reach their limits (Avola et al., 2022; Chadha et al., 2019; Zschech et al., 2019).

Consequently, it requires methods that are able to identify *any* kind of anomaly in a given instance without relying on pre-defined classes and labeled examples. This task can be realized with methods from the field of *unsupervised learning*. Such methods are typically not trained using any abnormal instances but rather focus only on normal instances. More specifically, they can learn a representation of a normal quality state from which deviations can be detected automatically. In related domains such as structural health monitoring (Bao et al., 2019) or medical diagnosis (Esteva et al., 2021), unsupervised models have already achieved promising results. Likewise, there have been notable developments in the area of unsupervised deep learning in recent years that seem promising for the context of industrial quality assurance. Respective methods can be roughly divided into reconstruction-based methods (e.g., Akçay et al., 2019; Gong et al., 2019; Perera et al., 2019) and representation-based methods (e.g., Defard et al., 2020; Roth et al., 2021; Yi & Yoon, 2020). However, the majority of existing methods have primarily been developed and evaluated with synthetic data collections and benchmark datasets under experimental laboratory conditions. Likewise, there is only limited evidence of how well unsupervised deep learning models perform in comparison to supervised counterparts in real-world applications.

As a remedy, we carry out an in-depth comparative evaluation study of three state-of-the-art unsupervised deep learning models for the task of anomaly detection in visual industrial quality assurance. More specifically, we consider the special case of detecting damaged vehicle labels during a quality inspection process of a European car manufacturer. The manufacturer was facing the challenge to detect any kind of previously unknown anomaly such as cracks, bubbles, and edge defects at an early stage since damaged labels on vehicles that have already been shipped lead to huge rework costs and can even cause the cars to be rejected at the importing country. For our comprehensive evaluation under real conditions, a rich dataset of registration-relevant labels was provided which is subject to varying lighting conditions and location-specific label layouts. By implementing three different types of unsupervised models and evaluating their detection performance comprehensively, we demonstrate that reliable results can be achieved that are even competitive with those of supervised counterparts which require pre-defined categories.

Our paper is structured as follows. Section 2 provides an overview about necessary foundations and related work. We then proceed to outline our research approach in Section 3 by presenting further details on the anomaly detection case, the implemented deep learning models as well as the experimental setup and evaluation procedures. Subsequently, we present our results in Section 4, followed by a discussion of our findings in Section 5. We conclude this work in Section 6.

2. Foundations and related work

2.1. Visual quality inspection and anomaly detection

Visual quality inspection plays a central role in manufacturing processes as it ensures that products meet certain aesthetic standards. This is important for a number of reasons. First and foremost, the visual appearance of a product is often closely tied to its quality. Consumers expect products to look a certain way, and if a product does not meet these expectations, it can reflect poorly on the company that produced it. Additionally, visual defects can sometimes indicate underlying problems with a product, such as structural issues or functionality problems. By conducting thorough visual inspections, companies can ensure that their products meet the necessary standards and provide a positive experience for their customers (Olimov et al., 2022; Zamora-Hernández et al., 2021).

Traditionally, visual quality inspections were carried out by human operators based on domain knowledge, heuristics, and handcrafted rules. However, manual quality inspections involving human labor are monotonous, time-consuming, and tiring tasks. Moreover, they are prone to errors due to human-related characteristics, such as subjective judgement, optical illusion, lack of concentration, and inclination to exhaustion. Another drawback is that human inspectors usually require time-consuming training and there is a risk of losing knowledge in the event of staff turnover (Olimov et al., 2022; Zheng et al., 2021).

Due to these limitations, modern quality inspection strategies increasingly rely on algorithmic approaches from the field of anomaly detection for automated defect detection (e.g., Alizadeh & Ma, 2021; Avola et al., 2022; Dang et al., 2021; Li, 2009; Olimov et al., 2022). In general, anomaly detection is concerned with the task of recognizing patterns in data that do not conform to a well defined notion of expected or normal behavior (Chandola et al., 2009; Pang et al., 2021). In the context of this work, normal instances are visual objects in the form of recorded images that are free of any damage. By contrast, anomalous instances are images that contain one or more damage types (e.g., cracks, scratches, cuts, etc.).

Depending on the type of data and the properties of the abnormal instances, the field of anomaly detection offers a wide range of techniques drawn from different research areas such as statistics, machine learning, information theory, and spectral theory (Chandola et al., 2009). Machine learning approaches have proven to be particularly beneficial for industrial contexts (Bertolini et al., 2021; Kang et al., 2020). Fueled by the widespread use of information technology and the collection of extensive sensor data, such approaches are able to iteratively learn from large amounts of problem-specific training data in order to capture complex patterns and hidden relationships that are relevant for the anomaly detection task (Janiesch et al., 2021). Because of this property, machine learning models have the advantage of improving over time by learning from an increasing number of available observations. In addition, it is possible to perform a high number of visual inspections in near real-time while ensuring an objective and consistent quality assessment (Kang et al., 2020).

The overall task of automatically detecting anomalies can be defined as either a supervised or an unsupervised learning task. In other words, known relationships from past observations are either given or not given during model training (cf. Section 2.3). Depending on the learning task, a variety of learning algorithms are applicable, such as support vector machines (SVMs), artificial neural networks (ANNs), clustering algorithms, Bayesian networks, etc. (Chandola et al., 2009). Among these examples, ANNs have turned out to be particularly promising for visual quality inspection in recent years because they can be applied for both learning types. At the same time, they are able to automatically discover useful feature representations in high-dimensional datasets that are beneficial for image-based anomaly detection (Pang et al., 2021; Yang et al., 2020).

2.2. Artificial neural networks and deep learning architectures

Inspired by information processing in biological systems, ANNs consist of multiple interconnected processing units, called neurons. These neurons transform incoming signals with an activation function and forward the processed signals to their successors weighted by the connections to them. The weights are the trainable parameters of an ANN, which get adjusted during the learning process (e.g., via backpropagation). Typically, neurons are organized into multiple, hierarchical processing layers that can be arranged into deep network architectures. This gives them the capability to process raw input data and automatically learn a feature representation that is relevant for the corresponding prediction task, which is commonly known as *deep learning* (LeCun et al., 2015). This capability makes ANNs more versatile and efficient than traditional methods that rely on handcrafted rules or advanced feature engineering approaches (Janiesch et al., 2021).

Furthermore, deep neural networks (DNNs) can be designed as different architectures to suit different data types, such as images with spatial dependencies (Kraus et al., 2020). Some important DNN architectures are explained in the following paragraphs.

Convolutional neural networks (CNNs). CNNs are a special form of ANNs that are mainly applied to computer vision-related tasks. CNNs perform a mathematical operation named convolution which uses the spatial information of neighboring pixels in images. The architectural design of CNNs allows them to automatically extract features at different levels of abstraction. Considering a CNN for classification, the first layers of the CNN extract very simple features (e.g., edges, corners). The following layers extract features of increasing abstraction levels and the last few layers resemble the features which are specific for each class. The final layer consists of neurons, indicating the class predicted for an image (Goodfellow et al., 2016). Special CNN architectures used from the models evaluated in this work are residual neural networks (ResNet) (He et al., 2015), networks from the visual geometry group (VGG) (Simonyan & Zisserman, 2015), and EfficientNets (Tan & Le, 2020).

Generative adversarial neural networks (GANs). GANs belong to the family of generative models. They aim to learn a probability distribution over a set of known training data to randomly generate new data samples following this learned distribution. To achieve this goal, two neural networks – a generator and a discriminator – are placed in a game theoretical scenario competing against each other. Thereby, the generator aims to generate instances which resemble instances of a given data set, i.e. fake instances. The discriminator tries to identify these fake instances. This said, both networks are trained together in a non-cooperative zero-sum game where one network's gain is another one's loss. The GAN is trained until the discriminator is no longer able to distinguish between original and generated samples (Janiesch et al., 2021).

Autoencoders. An autoencoder is a special type of ANN that aims to copy its input to its output. For this purpose, the autoencoder consists of two parts. The first part is an encoder, which compresses the input into a low-dimensional latent representation through multiple, in size decreasing hidden layers. Given this latent representation, a decoder – as the second part of an autoencoder – tries to reconstruct the original image through multiple, in size increasing hidden layers. Using this procedure, the autoencoder is forced to keep meaningful information in the latent representation and discard irrelevant, noisy information. While autoencoders are commonly applied as feature reduction algorithms (Goodfellow et al., 2016), they are also used in unsupervised learning, where they are, for example, trained on normal images only to find abnormal ones yielding a higher reconstruction loss (Pang et al., 2021).

2.3. Supervised and unsupervised models for visual anomaly detection

The research field at the intersection of anomaly detection and deep learning has gained increasing attention in recent years, resulting in diverse approaches that stem from different domains. Table 1 shows an excerpt of related work. For better readability, the number of datasets used in the publications are limited to three examples. In the following, we briefly summarize some of the most important developments.

As outlined before, the anomaly detection task can generally be phrased as either a supervised or an unsupervised learning task.[1] In *supervised learning*, the model uses known relationships in the form of ground truth data that represent different classes of normal vs. anomalous instances. The ground truths are expressed by annotations/labels that are usually provided by human domain experts (Sager et al., 2021).

In deep learning-based supervised anomaly detection, it is common to use deep CNNs such as ResNet, VGG, and EfficientNet to extract and learn discriminative features of an image and then classify these features as normal or anomalous based on the provided labels. The classification is usually done using a CNN with fully connected layers at the end of the network (Harl et al., 2022; Pang et al., 2021). A slightly different approach was pursued by Lopes and Alexandre (2020). Instead of using a single CNN, the authors use multiple pre-trained CNNs, which are fine-tuned on the specific dataset, to obtain a fused classification from the results of each CNN. Furthermore, the authors suggest to replace the classification head of the CNN with an alternative classifier found using AutoML functionalities. That is, instead of using a multilayer perceptron in the form of fully connected layers at the end of the CNN, an AutoML search procedure aims at finding a better performing classification head (e.g., XGBoost, random forest, generalized linear model, etc.).

Another alternative was proposed by Tao et al. (2018). The authors use a cascaded autoencoder network that transforms defect images into a pixel-wise prediction mask using semantic segmentation. Subsequently, the defect regions of the segmentation results are classified in a supervised manner using a compact CNN. Similarly, Tabernik et al. (2019) first use a segmentation network that performs a pixel-wise localization of defect patterns. Subsequently, the segmentation output as well as the features of the segmentation network are fed into a second network that performs the final binary image classification task.

In general, supervised anomaly detection models are able to achieve strong detection qualities because they can learn nuanced characteristics that distinguish anomalous instances from normal ones. While this is beneficial in domains where a large number of labeled instances exist, supervised approaches reach their limitations in highly standardized industrial processes. In such settings, deviations from normal conditions occur very rarely which results in highly skewed datasets. Thus, it is hard to find and annotate images of the underrepresented class (e.g., rare damage classes). Since artificially collecting and annotating anomalies is a time-consuming and costly endeavor (Sager et al., 2021), and in production settings it is usually not feasible to intentionally provoke any damage cases (Zschech et al., 2019), supervised learning is often not a viable option. Furthermore, it is often not possible to know a priori all the failure classes that may occur over time as a result of external disturbances, making it impossible to separate them in pre-defined classes.

These limitations can be addressed by *unsupervised learning* where models are trained without using known relationships between normal and anomalous instances. Instead, unsupervised models only use normal instances during training but need to identify anomalies in testing, which is also referred to as one-class classification or out-of-distribution detection (Yu et al., 2021). This overall task can be achieved in different ways where it is possible to distinguish between (i) reconstruction-based methods and (ii) representation-based methods (Zheng et al., 2022).

Reconstruction-based methods usually adopt some kind of generative architectures such as autoencoders (e.g., Gong et al., 2019), GANs (e.g., Perera et al., 2019), or a combination of both (e.g., Akçay et al., 2019) to generate or reconstruct input images and then assess the difference between input images and reconstructed images in order to identify anomalies. As such, these methods focus on encoding the manifold of normal images into a latent space, which is used to reconstruct them afterwards. Assuming that only normal images are present in the training data, the model is not able to correctly reconstruct anomalous instances during inference because they were not observed at the training stage. This is reflected by a higher anomaly score, which measures the reconstruction error between the input image and the reconstructed image (e.g., pixel-wise L2-norm) (Zheng et al., 2022).

[1] Some conceptualizations also offer a further distinction between fully supervised, semi-supervised, self-supervised, and fully unsupervised learning (e.g., Chandola et al., 2009; Pang et al., 2021). However, such a fine-grained distinction is beyond the scope of this paper.

Table 1
Overview of related work for visual anomaly detection.

Publication	Domain	Code (license)	Approach	Dataset
Harl et al. (2022)	Solar wafer	Official (no license)	Supervised	Own dataset
Tao et al. (2018)	Metallic surfaces	Not published		Own dataset, DAGM 2007
Tabernik et al. (2019)	Electrical commutators	Not published		Own dataset
Lopes and Alexandre (2020)	Dataset domain	Official (MIT)		DAGM2007
Huang et al. (2021)	Dataset domain	Not published		Fashion-MNist, CIFAR-10, Camelyon16
Yi and Yoon (2020)	Dataset domain	Unofficial (no license)	Unsupervised representation-based	MVTec
Ruff et al. (2018)	Dataset domain	Official (MIT)		MNIST, CIFAR-10, GTSRB
Reiss et al. (2021)	Dataset domain	Official (SRL)		CIFAR10, CatsVsDogs, MVTec
Yu et al. (2021)	Dataset domain	Unofficial (no license)		BTAD, CIFAR-10, MVTec
Defard et al. (2020)	Dataset domain	Official (MIT)		MVTec
Roth et al. (2021)	Dataset domain	Unofficial (Apache-2.0)		MVTec
Rudolph et al. (2020)	Dataset domain	Official (no license)		MVTec AD, Magnetic Tile Defects
Gong et al. (2019)	Dataset domain	Official (MIT)	Unsupervised reconstruction-based	MNIST, CIFAR-10, UCSD-Ped2
Perera et al. (2019)	Dataset domain	Official (Apache-2.0)		COIL100, fMNIST, CIFAR10
Akçay et al. (2019)	Dataset domain	Official (MIT)		CIFAR-10, UBA, FFOB

By contrast, *representation-based* methods follow a slightly different approach. In a first step, they apply different feature extractors to obtain discriminative features either from normal images (e.g., Rudolph et al., 2020; Ruff et al., 2018; Yu et al., 2021) or from normal image patches (e.g., Defard et al., 2020; Reiss et al., 2021; Roth et al., 2021; Yi & Yoon, 2020). Common feature extractors are deep CNNs that have been pre-trained on large datasets, such as ResNet, VGG, and EfficientNet trained on ImageNet. In a second step, the extracted features are used to learn a distribution of the normal features by using different estimation approaches. For example, some approaches that use discriminative features of a whole image utilize normalizing flows to project the extracted high-dimensional features into a standard normal distribution (e.g., Rudolph et al., 2020; Yu et al., 2021). During the estimation of the model, the discriminative features of the approaches using image patches only serve as distributions of normal images against which an image is compared. Finally, in a third step, an anomaly score is retrieved by calculating the similarity or distance between the features of a test image and the learned feature distribution. For the computation of the similarity, previous approaches are based on Mahalanobis distance (e.g., Defard et al., 2020) or apply a k-nearest neighbor (kNN) algorithm (e.g., Reiss et al., 2021; Roth et al., 2021).

Overall, the field has brought forth many promising approaches especially for unsupervised visual anomaly detection. Apart from innovative method developments, some studies also focused on comparative model evaluations to assess the merits and limitations of different models (e.g., Chadha et al., 2019; Siegel, 2020; Zheng et al., 2022). However, most approaches are solely developed and/or tested under experimental laboratory conditions, in which almost exclusively synthetic data collections and benchmark datasets were used. Therefore, a deeper model evaluation under real conditions is often missing. To narrow this gap, we contribute new insights with a comprehensive evaluation study based on a real-world case from a car manufacturer. In this way, we aim to comprehensively evaluate the detection performance of three state-of-the-art unsupervised models and examine whether they are capable of achieving results that can compete with those of supervised models.

3. Research approach

This section introduces our methods by which we detect damaged vehicle labels during a quality inspection process of a European car manufacturer. Fig. 1 illustrates our framework. In detail, we compare multiple state-of-the-art unsupervised methods. We also evaluate a state-of-the-art supervised method. In the following sections, we motivate our use case, describe our methods, and our experimental setup and evaluation procedure.

3.1. Case and data description

To carry out our research, we collaborated with a large European car manufacturer that produces vehicles in many different sites and locations around the world. Besides the focus on the actual manufacturing processes, the company has its own software development departments. They are responsible for accelerating the digitization of the entire company, developing industry-specific software solutions, and providing the group-wide IT infrastructure. The solutions and services of the software development departments are used in many areas, for example internally for production and administrative processes as well as externally for customer services and convenience features. Therefore, the developed solutions have a central importance for delivering high-quality services and applications.

To improve resource-efficient production processes, the car manufacturer is currently developing a cloud-based infrastructure for industrial applications together with one of the biggest cloud providers and other partners in the field of manufacturing. The cloud is to be made available as a uniform and reusable architecture for all production sites of the company, aiming to integrate all available data along the global supply chain for the comprehensive optimization of the production processes.

In our project, we were in contact with one of the software development departments which is specialized in creating solutions for production and logistics. In the context of the cloud platform, the department has the responsibility to develop reliable industrial applications and offer significant support for the platform development itself. One sub-field in this area is the development of industrial computer vision applications for the automation of quality assurance processes. In cooperation with two production sites of one of the premium brands of the company, the department is working on an application to improve and automate the inspection of registration-relevant labels in the vehicle during production. The inspection is either directly carried out in the assembly line or in the rework area. The labels are checked for presence and the correctness of layout, position, content, and reading direction. In order to fully automate such a quality inspection, a check for damages to the label is essential. The aim of this work is to provide models for detecting anomalies to these registration-relevant labels to address this requirement.

In general, there are different types of registration-relevant labels such as labels for the vehicle identification number (VIN label), which serve to uniquely identify a vehicle, or tire pressure labels, which provide information about the recommended air pressure of car tires. Such labels are manually attached to the car body by employees during the assembly process. In this study, we primarily focus on VIN labels as they are associated with large costs in case of any deviations such as erroneous placements or damages. The VIN label contains information about the VIN itself (i.e., a world manufacturer code, a vehicle descriptive part, and a consecutive number) as well as further country-specific

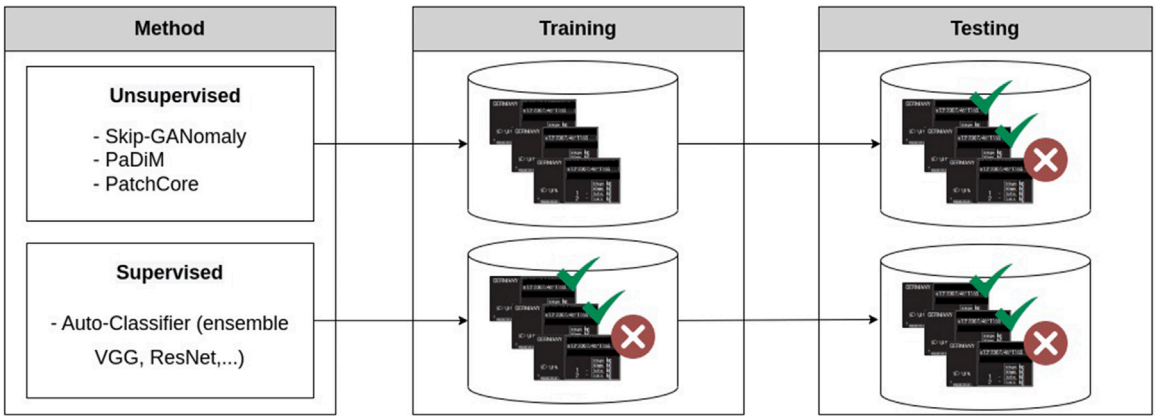

Fig. 1. Framework of our methods. Highlighted is the difference between supervised and unsupervised approaches in their training, in which supervised methods require an annotated dateset of images.

Fig. 2. Examples of undamaged (left column) and damaged (crack and corner damage; right column) VIN labels from Europe (top row) and Korea (bottom row).

data. The large variety of country- and vehicle-specific information on the labels leads to a high level of complexity when checking for anomalous patterns. Fig. 2 provides some examples of undamaged (left column) and damaged (right column) VIN labels. Due to the relevance for the registration of a vehicle, it is of great importance that the labels are free of any damage. Damaged labels can result in a vehicle, which is to be exported to another country, having to be reattached in a time-consuming process or even being rejected and thus not being able to be imported. Reliable and early detection of damage is therefore of utmost importance.

For the automated inspection of registration-relevant labels, they are photographed both on the assembly line by stationary cameras and during rework by mobile devices. The two imaging situations differ primarily in the lighting environment as well as camera angles and distances, which remain constant for stationary cameras and vary for the mobile imaging situation. This varying lighting environment poses a challenge to automated anomaly detection with machine learning models. Another challenge is the big variety of training data, since there are generally many different country-specific layouts for VIN labels without following a certain structure. Therefore, anomaly detection models must either be generic enough not to detect differences in layout as anomaly, or multiple models must be trained for one layout each.

To conduct our experiments, we collected a variety of real-world images covering VIN labels from different vehicles of the production sites from our collaboration partner. More specifically, damaged and undamaged images were taken from stationary cameras and smartphone cameras. However, since there was only a limited amount of damages to VIN labels in the production process, anomaly cases had to be constructed artificially. Please note that, although our study is particularly concerned with the use of unsupervised models that do not require any damaged VIN labels, a high proportion of damaged examples was necessary in order to properly evaluate the performance of the unsupervised models. For this purpose, flawless VIN labels were selected and damaged by physically inducing various defect types, such as cracks, bubbles, and edge defects. Subsequently, the damaged labels were glued on surfaces of test vehicles for photography to receive realistic images of anomaly cases. Overall, this resulted in a collection

of 2703 images of undamaged VIN labels and 970 images of damaged VIN labels.[2]

The average resolution of the original images was 1347 × 723. However, to obtain a unified dataset, the relevant VIN label sections were cropped out, standardized, and reduced to a unified size. To detect and extract VIN labels from the original camera recordings, we trained a Mask R-CNN (He et al., 2018) with the aim of segmenting the labels. The segmentation results in the form of masks were subsequently cut out and warped so that the VIN labels fit into a predefined rectangular structure for automated image processing by the deep learning models. All images in the dataset were standardized, so that the pixel distribution of each image is standard Gaussian. As a result, the image brightness variation caused by the different recording situations of the images was reduced. In addition, every image was resized to 256 × 256 pixels. All VIN labels in our experiments were similar in the way that they share a rectangular structure and have a black background with white lettering. Beyond that, however, there are also significant differences stemming from location-specific layouts as a result of covering seven main export countries and one class covering all remaining countries named "rest of the world". Depending on the country to which a vehicle is to be exported, a different layout is selected (cf. Fig. 2).

3.2. Selection of anomaly detection models

For the identification of suitable models for detecting anomalies in image data, we performed a structured literature search. More specifically, a hermeneutic approach was applied consisting of three cycles (Boell & Cecez-Kecmanovic, 2014) using the databases *IEEE Xplore*, *ScienceDirect* and *arXiv*. The search terms were applied to title, abstract, and keywords. In a first cycle, a general overview of the problem domain of anomaly detection was established using the search term *anomaly detection*. Based on a large result set (9694, 8764, and 1830 hits in the respective databases), crucial concepts could be derived in order to refine the field. Thus, in a second cycle, the search term was extended to *anomaly detection AND computer vision AND NOT video AND NOT time series*. As a result, the amount of relevant literature could be reduced significantly to 290, 317, and 326, respectively. To further specialize this result set for the identification of problem-related methods, the term *anomaly detection* was replaced by *defect detection*, resulting in 434, 549, and 74 search hits, respectively.

Based on all three cycles, 90 publications could be identified that propose different kinds of machine learning and deep learning models for defect detection based on image data. An overview of all 90 publications, which we classified according to their type of learning approach, can be found in Appendix B.1 of the supplementary material. In order to identify suitable candidates for implementation and evaluation purposes, we defined several selection criteria. More specifically, the following five criteria were defined: (i) general suitability for the case at hand, (ii) reusability for other use cases of the collaboration partner, (iii) use of a deep learning model for better comparability, (iv) availability of source code to facilitate implementation, (v) availability of MIT-license as a prerequisite for commercial usage by the software development department. Based on the criteria, we ultimately selected four different approaches for an in-depth examination through a comparative evaluation study. More specifically, we chose three competing unsupervised models and one supervised model as a baseline approach. The selected and implemented models are briefly described below. A more detailed description of each approach, including graphical illustrations, can be found in Appendix B.2 of the supplementary material.

Skip-GANomaly. The first model is an unsupervised reconstruction-based approach, called Skip-GANomaly (Akçay et al., 2019). As an unsupervised model, it is trained on normal (undamaged) images only and tested on both normal and anomalous (damaged) images in order to reasonably determine the decision threshold for the anomaly score. The general architecture of Skip-GANomaly is similar to that of GANs, consisting of two competing neural networks. The first one is a generator network based on an autoencoder and the second one is a discriminator network with a similar structure as the encoder with additional layers added to determine a classification. Based on the GAN-like architecture, the generator tries to copy the original image as well as possible to its output. The discriminator's task is to distinguish reconstructed images from the original ones. As the generator gets better at reconstructing, the discriminator must get better at continuing to classify the images correctly. However, the generator will fail at reconstructing an anomalous image making the discriminator determine it as reconstruction. The anomaly score is then calculated as the weighted sum of the reconstruction loss (L2-norm) and the latent-representation loss (L2-norm).

PaDiM. The second selected model is an unsupervised representation-based model, called PaDiM (Defard et al., 2020). Unlike Skip-GANomaly, PaDiM does not use the reconstruction loss of an image, but the similarity of embedding vectors. An embedding vector is equal to the concatenation of feature maps from different hidden layers of a CNN. For this purpose, PaDiM uses several pre-trained CNN backbone networks, such as ResNet18 (He et al., 2015), WideResNet50 (Zagoruyko & Komodakis, 2017), and EfficientNetB5 (Tan & Le, 2020). The training process of PaDiM differs from the training of a normal neural network in that no weights of the model are adjusted to minimize an error function during training. Instead, only the feature maps from the forward pass of the train images that go through the pre-trained CNN are used.

PatchCore. The third selected model is another unsupervised model following a representation-based approach, called PatchCore (Roth et al., 2021). As with PaDiM, a pre-trained CNN backbone network (WideResNet50) is used to extract embedding vectors. However, in contrast to PaDiM, the embedding vectors are first stored in a so-called memory bank. To reduce computational effort, the memory bank is subsampled by applying a k-center-greedy algorithm (Wolf, 2011). Since the subsampling is computationally expensive, the original memory bank is reduced in dimensionality by a random projection (Van Der Maaten et al., 2009). The result of this procedure is called embedding coreset. To determine a test image as normal or anomalous, its embedding vectors are extracted in the first step. Afterwards, k distances to the vectors of the embedding coreset are calculated using the k-nearest neighbor algorithm (Cover & Hart, 1967). Given these k distances, an anomaly score is calculated for each image path using the distance to the nearest neighbor weighted by the other k distances.

Auto-Classifier. Besides all three unsupervised models, we also selected a supervised approach to include a baseline model for a competitive comparison. More specifically, we chose the Auto-Classifier model proposed by Lopes and Alexandre (2020), which combines a CNN-fusion approach with an AutoML procedure. In the fusion approach, several pre-trained CNN architectures are retrained. After that, the test results of all CNN architectures are merged ("fused") together, weighted by an evaluation metric (the better an individual model performs, the more weight their vote gets), yielding a single classification result, which can be interpreted as an anomaly score. Several CNNs from the VGG- and ResNet-families are used as backbones (i.e., VGG11, VGG16, VGG19, ResNet18, ResNet34, ResNet50, ResNet101). The second part of this approach (i.e., AutoML) optimizes the classification part of the best individual CNN backbone for the use case by replacing the last part of the classifier with a new one, determined by an automated machine learning procedure (LeDell & Poirier, 2020).

[2] Please note that for reasons of confidentiality, the expected distribution of damaged and undamaged VIN labels cannot be reported.

3.3. Experimental setup and evaluation procedure

To carry out our comparative evaluation study, we performed a series of computational experiments. More specifically, we applied all previously described models for the task of anomaly detection on the VIN label dataset consisting of 2703 undamaged and 970 damaged VIN labels. For the technical realization of each model, we used the publicly available implementations from GitHub (Skip-GANomaly[3], PaDiM[4], PatchCore[5], Auto-Classifier[6]). To provide a fair comparison and to be able to make any necessary adjustments to the existing source code, we integrated all implementations into a shared environment to run the experiments under the same conditions. Our environment was based on Python (version 3.8) as programming language, PyTorch (version 1.9) as machine learning framework, Scikit-learn for machine learning tools, and Matplotlib for visualizations. The training and inference runs of all models were performed on a server instance of Amazon Web Services Elastic Cloud Computing (AWS EC2 instance of type p2.xlarge) based on an Intel Xeon Processor (Broadwell E5-2686 v4) with four CPU cores and 61 GB RAM and a NVIDIA Tesla K80 with one GPU core and 11 GB GPU memory. Our customized source code with all implemented models can be found on GitHub[7].

Regarding the choice of hyperparameters, we adopted the recommended default values from the original articles, which have been thoroughly investigated for the most critical parameters by the respective authors in comprehensive ablation studies. Additionally, we performed several ablation studies ourselves to verify the robustness of the different models (cf. Section 4.3). An overview of the implemented (and ablated) hyperparameter values can be found in Table 9 in Appendix A.1.

To assess the detection qualities of the models, the dataset was split into train, validation, and test sets using stratified random sampling. The train set is used to train the respective models. The validation set is used to find the optimal train epoch for the supervised approach and the optimal decision threshold for all models. Finally, the test set is used to measure the models' out-of-sample performances for the anomaly detection task. For the supervised models, the dataset is split into 70% train (74% damaged vs 26% undamaged), 15% validation (74% damaged vs 26% undamaged) and 15% test (74% damaged vs 26% undamaged). For the unsupervised models, the proportions of the train and validation splits are changed as the unsupervised models do not require any damaged samples. In this case, the dataset consists of 52% train (100% undamaged), 33% validation (33% damaged vs 67% undamaged), and 15% test samples (74% damaged vs 26% undamaged). Table 2 provides an overview of the exact distribution among the individual splits of the dataset.

For the reproducibility of the model results, we were running all our experiments five times with different seeds, which ensures that all algorithms and procedures based on random generators behave the same. This applies, for example, to the random sampling of the three data splits. Additionally, the use of multiple seeds allows us to check how stable the results of the different models are under different weight initialization for the neural networks in order to avoid outlier results due to local optima.

For the quantitative assessment of the detection performance, we calculated multiple evaluation metrics. The metric primarily used for our comparative evaluation is the *area under the receiver operating characteristic (AUROC)*. Mathematically, the receiver operating characteristic measure is a graph, which shows the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The area under this graph is then called *AUROC* and is always between 0 and 1, with 1 denoting the best performance. Since this metric is invariant to the decision threshold and considers both classes (normal and anomalous) equally, it allows a generic comparison of all approaches. Additionally, the metric is used to find an optimal threshold, which is further used to calculate other metrics, such as $Recall$, $Precision$, $F1\ Score$ and confusion matrices. Mathematically, for a model f, dataset D with images d and corresponding labels y, and optimal threshold t, these metrics are defined as

$$\text{Recall} = \frac{\sum_{d,y\in D} \mathbb{1}_{f(d)>t} \times \mathbb{1}_y}{\sum_{d,y\in D} \mathbb{1}_y} \tag{1}$$

$$\text{Precision} = \frac{\sum_{d,y\in D} \mathbb{1}_{f(d)>t} \times \mathbb{1}_y}{\sum_{d,y\in D} \mathbb{1}_{f(d)>t}}, \tag{2}$$

where $\mathbb{1}$ describes the indicator function. The $F1\ Score$ is calculated as the harmonic mean between $Recall$ and $Precision$.

Beyond that, we calculated the $Recall_{Pr=x}$ metric, as it reflects specifications from the industry. It is a special case of $Recall$, which is calculated while maintaining a fixed precision value (certainty). Therefore, the decision threshold, which determines whether a score is normal or abnormal, is set such that a fixed $Precision$ is achieved. The corresponding $Recall$ is the value of the $Recall_{Pr=x}$ metric. Mathematically, the $Recall_{Pr=x}$ is defined as

$$Recall_{Pr=x} = \max_{t\in[0,1]} \frac{\sum_{d,y\in D} \mathbb{1}_{f(d)>t} \times \mathbb{1}_y}{\sum_{d,y\in D} \mathbb{1}_y} \tag{3}$$

$$s.t.\ \frac{\sum_{d,y\in D} \mathbb{1}_{f(d)>t} \times \mathbb{1}_y}{\sum_{d,y\in D} \mathbb{1}_{f(d)>t}} \geq x, \tag{4}$$

thus, we maximize the $Recall$ under the constraint that a $Precision$ of x must be achieved.

The fixed precision values are set to 99.6%, 95%, and 90%. The value 99.6% is the required safety threshold from our industrial collaboration partner for a model to be deployed in a production plant. For the calculation of all metrics, the anomalous cases are used as the positive class ($y = 1$).

Besides the assessment of the detection performance, we also included further metrics for our comparative evaluation. On the one hand, we measured the time each approach requires for model training, since training time on GPU is generally costly. On the other hand, only a limited amount of time is allowed on the assembly line to perform the anomaly detection task, which is why we also measured inference times during model application. Additionally, we also performed a qualitative assessment of the model results by systematically examining their misclassifications (i.e., false positives (FP) and false negatives (FN)). For this purpose, we looked into the confusion matrix as a starting point and subsequently examined the misclassified images more closely in order to draw conclusions about the cause of the misclassification. Thus, possible strengths and weaknesses of the respective models can be identified.

The main evaluation takes place on the whole dataset, which is considered as our main experiment. Beyond that, we also conducted further experiments to evaluate how the selected models perform under slightly changed conditions. Thus, in a first variation of the experiment, the images of the dataset are rotated so that the VIN labels are readable from left to right and from top to bottom to decrease the variance of the images. In a second variation, these direction-corrected VIN labels are separated into images from stationary and mobile imaging environments. In the stationary separation, however, the damaged VIN labels of the mobile imaging environment continue to serve as anomalous images. To align these two sets, all images are converted to grayscale.[8] By contrast, no changes were necessary in the dataset

[3] https://github.com/samet-akcay/skip-ganomaly
[4] https://github.com/Pangoraw/PaDiM
[5] https://github.com/hcw-00/PatchCore_anomaly_detection
[6] https://github.com/VascoLopes/AutoClassifier
[7] https://github.com/SDJustus/CAIE-anomaly-detection

[8] This step was done because all stationary images were only available in grayscale, whereas a large proportion of the damaged VIN labels were also taken in color images.

Table 2
Separation of the dataset into training, validation, and test split.

	Total		Training		Validation		Test	
	Undamaged	Damaged	Undamaged	Damaged	Undamaged	Damaged	Undamaged	Damaged
Supervised	3673 (100%)		2571 (70%)		550 (15%)		552 (15%)	
	2703 (74%)	970 (26%)	1892 (74%)	679 (26%)	405 (74%)	145 (26%)	406 (74%)	146 (26%)
Unsupervised	3673 (100%)		1892 (52%)		1229 (33%)		552 (15%)	
	2703 (74%)	970 (26%)	1892 (100%)	–	405 (33%)	824 (67%)	406 (74%)	146 (26%)

Table 3
Evaluation results of the anomaly detection performance based on the test set.

Model	Backbone	*AUROC*	*F1 Score*	$Recall_{Pr=0.996}$	$Recall_{Pr=0.95}$	$Recall_{Pr=0.9}$
Auto-Classifier	ResNet50	1.000 ± 0.000	0.995 ± 0.005	0.995 ± 0.009	1.000 ± 0.000	1.000 ± 0.000
	Fusion	1.000 ± 0.001	0.995 ± 0.006	0.792 ± 0.443	1.000 ± 0.000	1.000 ± 0.000
	AutoML	1.000 ± 0.000	0.992 ± 0.003	0.978 ± 0.022	1.000 ± 0.000	1.000 ± 0.000
Skip-GANomaly	–	0.930 ± 0.006	0.757 ± 0.009	0.130 ± 0.081	0.214 ± 0.166	0.275 ± 0.160
PaDiM	EfficientNetB5	0.992 ± 0.002	0.911 ± 0.009	0.438 ± 0.240	0.884 ± 0.031	0.940 ± 0.020
	ResNet18	0.978 ± 0.003	0.874 ± 0.010	0.230 ± 0.092	0.486 ± 0.206	0.790 ± 0.048
	ResNet50	0.982 ± 0.005	0.897 ± 0.017	0.205 ± 0.143	0.697 ± 0.112	0.841 ± 0.080
	WideResNet50	0.984 ± 0.005	0.893 ± 0.033	0.318 ± 0.113	0.712 ± 0.136	0.867 ± 0.087
PatchCore	WideResNet50	0.996 ± 0.002	0.948 ± 0.015	0.451 ± 0.245	0.941 ± 0.033	0.986 ± 0.017

for the mobile imaging environment. Finally, in a third variation, we decreased the variation resulting from different layout specifications. For this purpose, we considered only one particular VIN label layout (i.e., "rest of the world" layout), for which we used the pre-processed, direction-corrected images. An overview of the data distribution for the creation of the different data splits can be found in Table 10 in Appendix A.2.

Overall, all the experiments with changed conditions represent image set constraints and pre-processing steps that are also feasible in the context of the use case with manageable effort. Therefore, the experiments were intended to reveal whether such refinements and context-specific models are necessary to significantly boost the performance, or whether the initial global models from the main experiment can already achieve a satisfactory level of detection performance.

4. Results

In this section, we present the results of our computational experiments. As outlined above, all models are trained with five different seeds to ensure reproducibility and stability of the results. For improved readability, only the mean values and standard deviations of the five runs are reported. The evaluation of the anomaly detection performance is based on the test set which has not been used for model training. This test set contains a total of 552 images with 406 being undamaged and 146 being damaged VIN labels. The results are summarized in Table 3. From the standard deviation, which is constantly below 0.01 for *AUROC*, it can be inferred that the model training is stable across all models.

As expected, the best detection performance is achieved by the supervised Auto-Classifier approach as it is trained on pre-defined target classes with labeled instances. More specifically, the best CNN backbone from the implemented ResNet- and VGG-families turned out to be a ResNet50, which is the only individual backbone network reported in Table 3 for better readability.[9] It achieved an average *AUROC* value of 1.0, which corresponds to a perfect detection performance. For this CNN, three of the five runs have an *AUROC* value of 1.0 (i.e., no misclassifications). Considering the $Recall_{Pr=0.996}$, the model finds 99.5% of anomalies with a certainty of 99.6% percent (with a standard deviation of less than 1%). Due to its superiority, the ResNet50 also formed the basis for the AutoML approach. However, the results are slightly worse than those of the standalone ResNet50. Thus, it did not add any value to the case at hand. A similar result could also be observed for the CNN fusion approach, which fuses the classification results of all individual ResNet- and VGG-backbones. Hence, it also performs slightly worse than the standalone ResNet50.

Table 4
Evaluation results of training and inference times for a single run.

Model	Backbone	Train time [h]	Inference time [ms]
Auto-Classifier	ResNet50	1.401	17.48
	Fusion	15.528	199.96
	AutoML	2.433	10.65
Skip-GANomaly	–	4.935	150.39
PaDiM	ResNet18	0.211	57.89
	ResNet50	0.236	111.98
	WideResNet50	0.241	119.96
	EfficientNetB5	0.248	103.62
PatchCore	WideResNet50	44.491	1818.52

Looking at the results of the unsupervised models, the Skip-GANomaly performs worst among the three candidates with an *AUROC* value of 0.93. Although the *AUROC* value suggests a strong model performance as well, the poor performance becomes clearer when considering the $Recall_{Pr=0.996}$ metric, which is at 13%. Put differently, only 13% of all anomalies are found if the *Precision* is set to be at least 99.6%.

By contrast, the PaDiM model performs very well with the best performing backbone network – an EfficientNetB5 – considering the fact that it is an unsupervised model. The PaDiM model leads to an average *AUROC* value of 0.992. As expected, the choice of the CNN backbone is critical to performance. Looking at $Recall_{Pr=0.996}$, it can be seen that for the EfficientNetB5 backbone, around 44% of the anomalies are found by the model for a given minimum *Precision* of 99.6%, while for example the ResNet18 backbone finds only 23% of the anomalies for the same *Precision* threshold.

The best unsupervised approach with an *AUROC* value of 0.996 is the PatchCore approach. For PatchCore, the $Recall_{Pr=0.996}$ metric of 45% is slightly better than the same metric of the PaDiM approach. From a certainty of 95%, a stable *Recall* (standard deviation of 3%) of 94% is seen with the PatchCore approach. For comparison, only about 88% of anomalies are found by PaDiM while maintaining the same level of certainty.

[9] The evaluation results of the alternative backbone networks can be found in Table 11 in Appendix A.3.

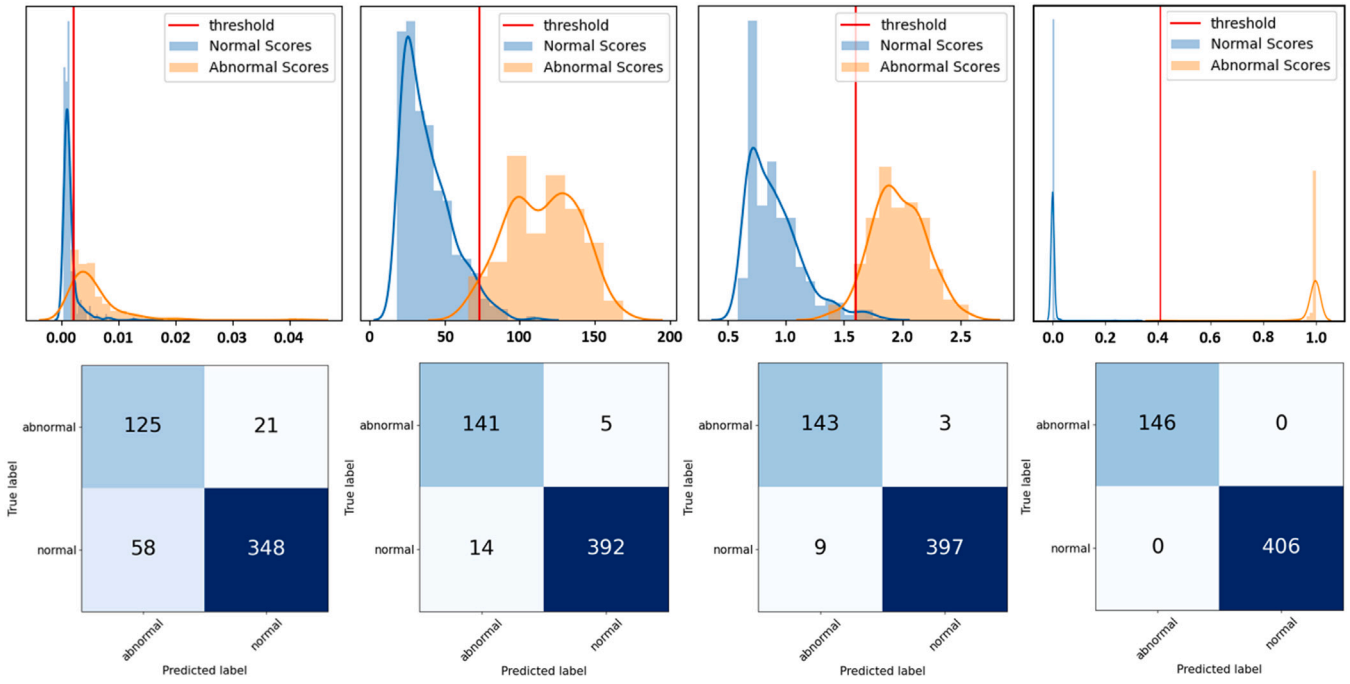

Fig. 3. Anomaly score distribution of Skip-GANomaly, PaDiM-EfficientNetB5, PatchCore, and Auto-Classifier-ResNet50 (from left to right).

The respective training and inference times can be taken from Table 4. First, it can be seen that the training and inference times for the Skip-GANomaly approach are in the middle range for the models. The inference takes the third longest time per image at about 150 ms - after the inference time of the PatchCore and CNN fusion approaches. The PaDiM model has the shortest training time, because only the embedding vectors are formed and then the mean value and the covariance of these vectors are calculated. Both training and inference time scale with the size of the backbone CNN and the number of embedding vectors used.

By far the most training time (almost two full days) is required by the PatchCore model, although it has a similar training procedure to the PaDiM model. This is due to the minimax facility location problem, which is NP-hard and takes 99% of the training time. The inference time is also by far the longest (about 1.8 s). This is due to the k-nearest neighbor search, which must compute all k distances from the inference image to the embedding vectors from the training coreset. After the coreset subsampling, the training set includes 484,352 embedding vectors.

For completeness, we also report the training and inference times of the supervised approach. In general, it can be observed that the individual CNNs have the shortest inference times as represented by the results for the ResNet50 approach. Further, it is noticeable that the AutoML approach is faster than the ResNet50, whose feature mapping in the last convolutional layer is the starting point for the classifier found by the AutoML procedure. The fusion approach, on the other hand, shows the longest times as they are calculated by the sum of the individual CNNs.

4.1. Qualitative assessment

For the qualitative assessment, the results of the model trainings are examined in more detail. For this purpose, we looked into the model results from the seed with the largest average *AUROC* value over all five runs. For an illustrative overview, we summarize the confusion matrices and anomaly score distributions for each model in Fig. 3. Furthermore, we provide some misclassification examples including the anomaly mappings (except for Skip-GANomaly) in Fig. 4.

Skip-GANomaly. It is noticeable that the model both misclassifies very often and cannot create a clear separation of the anomaly scores. Further, the anomaly scores lie in a very small interval. Since the anomaly score in this model equals the sum of the squared reconstruction error and the squared difference of the generator and discriminator bottle necks, the small interval of the anomaly scores indicates that the model reconstructs anomalous images approximately as well as normal images. Based on this fact, the model performs much worse than its competitors by showing the highest misclassification rate of 14.3%. Broken down, this rate corresponds to an FPR of around 10.5% and an FNR of around 3.8%. Looking at the misclassified examples in Fig. 4a, it can be assumed that the model has primarily learned to classify large colored areas as anomalous. The reason for this assumption is that, in this setting, all of the misclassified damaged VIN labels, although showing large damage patterns, were taken on a colorless background. In addition, around 60% of the misclassified undamaged VIN labels were taken on a colored background, which is still visible after cropping the image. The other 40% had lighting conditions, were cropped askew, or showed blurred sections.

PaDiM. For PaDiM, the model training with the EfficientNetB5 as a CNN backbone yields both significantly less misclassified VIN labels, as well as better separated anomaly scores (see second column in Fig. 3). In total, the model yields a misclassification rate of 3.4% (consisting of 2.5% FPR and 0.9% FNR). This corresponds to less than a quarter of the misclassifications that were made by Skip-GANomaly. Looking at the first three columns in Fig. 4b, it can be seen that the VIN labels are incorrectly classified due to unfavorable lighting situations (first two columns) or due to a barely visible scratch (third column). The columns four to six illustrate that the misclassified damaged VIN labels only consist of very small damages. However, in all images (except for the VIN label of the fifth column), the model correctly recognizes the anomalous regions. Similarly, it can be seen in Fig. 3 that the identified decision threshold for classifying a VIN label as an anomaly is only slightly higher than the threshold required to correctly detect all anomalies.

PatchCore. As the best-performing unsupervised model, the PatchCore model yields a misclassification rate of 2.2% (1.6% FPR and 0.6%.),

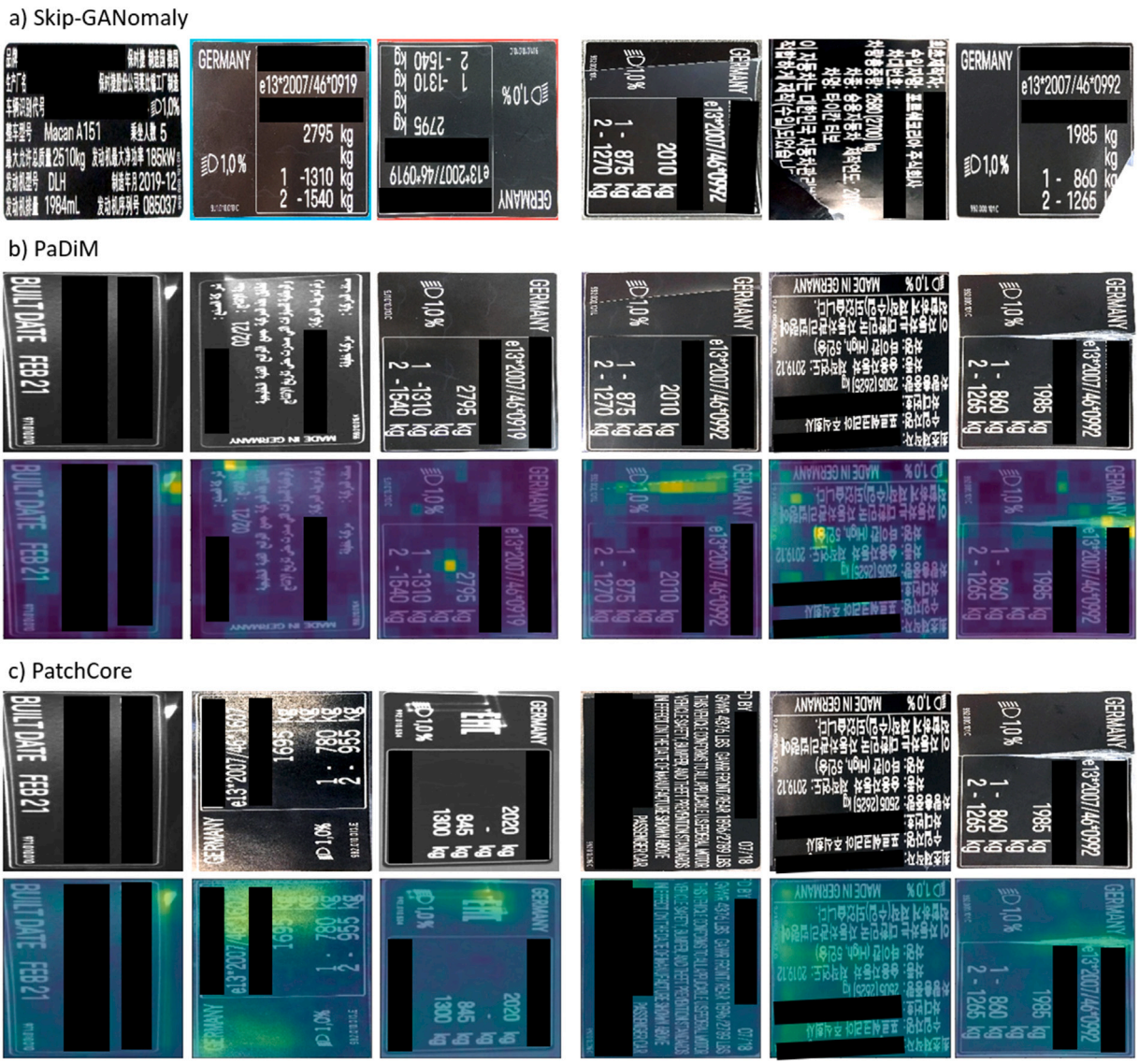

Fig. 4. False positives (first three columns) and false negatives (fourth to sixth column) of Skip-GANomaly (a), PaDiM (b), and PatchCore (c). Confidential information was obscured.

which corresponds to an error reduction of about 37% compared to PaDiM. Compared to Skip-GANomaly, the misclassification rate could even be reduced by about 85%. Fig. 3 (third column) shows that the model achieves a very good separation of anomaly scores given the fact that it is an unsupervised model. The misclassified undamaged VIN labels are shown in Fig. 4c (column one to three). The first misclassified VIN label is the same made by the PaDiM approach. The second misclassification was due to a reflection, and the third label shows the position laser used to correctly place the label in the image. The anomalous region of the first misclassified damaged label (column four) is correctly detected by the approach. With a lower decision threshold for the anomaly score, the image would have been correctly classified. The last two of the three labels misclassified as undamaged VIN labels are the same made by the PaDiM approach.

Auto-Classifier. The three parts of the Auto-Classifier model achieve the best separation of anomaly scores, as shown by the corresponding figures. As such, the models have by far the fewest misclassifications. Thus, the Auto-Classifier meets the expectation that it classifies as a supervised approach better than the unsupervised approaches. As mentioned above, the CNN-Fusion and AutoML parts of the Auto-Classifier approach do not add any value to the ability to correctly classify VIN labels of the ResNet50 as a standalone CNN, thus they are not evaluated here. For qualitative model assessment, the model generated using the best performing seed was selected. Fig. 3 (fourth column) thus shows a perfect confusion matrix and a perfect anomaly score distribution. Since no misclassifications are made by the approach, the Auto-Classifier does not occur in Fig. 4.

4.2. Results of the additional experiments

The experiments done in this section serve both to increase the accuracy and to further compare the unsupervised approaches. The results are summarized in the Tables 5–8. The supervised model is not examined further because it already made no misclassifications in the default configuration.

In the first experiment (see Table 5), the VIN labels are aligned in reading order to reduce the variance of the dataset so that the models only have to learn one reading order. All three unsupervised models increased their anomaly detection performance with this experiment. Even though this is an expected result, it is a valuable insight to establish such a configuration step as a standard measure and to tailor the underlying image collection system accordingly. Since this configuration has improved the performance of all models, while the number of

Table 5
Anomaly detection results based on images with corrected layout direction.

Model	*AUROC*	*F1 Score*	$Recall_{Pr=0.996}$	$Recall_{Pr=0.95}$	$Recall_{Pr=0.9}$
Skip-GANomaly	0.952 ± 0.010	0.790 ± 0.035	0.218 ± 0.156	0.301 ± 0.156	0.426 ± 0.245
PaDiM	0.992 ± 0.001	0.917 ± 0.013	0.542 ± 0.116	0.878 ± 0.040	0.940 ± 0.042
PatchCore	0.997 ± 0.002	0.961 ± 0.015	0.497 ± 0.406	0.974 ± 0.024	0.997 ± 0.004

Table 6
Anomaly detection results based on images from stationary camera.

Model	*AUROC*	*F1 Score*	$Recall_{Pr=0.996}$	$Recall_{Pr=0.95}$	$Recall_{Pr=0.9}$
Skip-GANomaly	0.995 ± 0.006	0.957 ± 0.022	0.856 ± 0.197	0.967 ± 0.023	0.973 ± 0.026
PaDiM	0.995 ± 0.002	0.965 ± 0.007	0.285 ± 0.269	0.973 ± 0.014	0.999 ± 0.003
PatchCore	0.998 ± 0.001	0.980 ± 0.011	0.538 ± 0.189	0.999 ± 0.003	1.000 ± 0.000

Table 7
Anomaly detection results based on images from mobile camera.

Model	*AUROC*	*F1 Score*	$Recall_{Pr=0.996}$	$Recall_{Pr=0.95}$	$Recall_{Pr=0.9}$
Skip-GANomaly	0.675 ± 0.040	0.581 ± 0.189	0.068 ± 0.070	0.126 ± 0.136	0.193 ± 0.159
PaDiM	0.982 ± 0.008	0.946 ± 0.026	0.423 ± 0.277	0.974 ± 0.019	0.995 ± 0.008
PatchCore	0.978 ± 0.008	0.952 ± 0.010	0.212 ± 0.175	0.973 ± 0.018	0.997 ± 0.004

Table 8
Anomaly detection results based on images from a single layout.

Model	*AUROC*	*F1 Score*	$Recall_{Pr=0.996}$	$Recall_{Pr=0.95}$	$Recall_{Pr=0.9}$
Skip-GANomaly	0.910 ± 0.022	0.769 ± 0.063	0.167 ± 0.086	0.172 ± 0.094	0.206 ± 0.122
PaDiM	0.995 ± 0.004	0.970 ± 0.016	0.581 ± 0.363	0.972 ± 0.017	0.989 ± 0.017
PatchCore	0.997 ± 0.001	0.974 ± 0.006	0.700 ± 0.242	0.986 ± 0.014	1.000 ± 0.000

images of both classes has remained the same, the direction-corrected dataset also serves as the basis for the following experiments.

In the next experiment, the VIN labels are separated according to the recording situation. As a result, the performance of the models for the stationary images increases (see Table 6). This increase is most evident in the Skip-GANomaly model, which has almost the same performance as the other two approaches. Thus, all models classify on exclusively stationary images better than in the previous experiment. However, as expected, the performance is much worse when exclusively focusing on images from mobile cameras (see Table 7). By far the worst model is the Skip-GANomaly model, which loses more than 0.25 *AUROC* points compared to the direction-corrected dataset. It is also striking that the overall best approach of this experiment is the PaDiM approach as opposed to the previously dominating PatchCore model.

In the last experiment, the dataset is separated by different label layouts. Table 8 lists the results of the "rest of the world" layout. Since the $Recall_{Pr=0.996}$ metric of the PatchCore approach reaches an excellent result of 70%, the model can already be used for detecting anomalies in a production setting. Thus, overall it can be stated that the PatchCore model provides the best results in the stationary recording situation for single label layouts, and the PaDiM model provides the best results in the mobile situation for single label layouts.

In summary, the following differences in terms of advantages and disadvantages can be concluded. Skip-GANomaly proved to be less suitable for anomaly detection when high industrial safety requirements have to be met. The advantages of the PaDiM model are mainly in the inference time, which, in contrast to the PatchCore model, is independent of the size of the training dataset and is faster than that of the PatchCore model. Moreover, in datasets with a large variance, the PaDiM approach is preferable in some cases. Beyond that, the PatchCore model was found to be the most preferable choice over the alternative models in almost all use cases. With sufficient refinement of the datasets, the PatchCore approach can even achieve perfect classification results and thus keep up with supervised approaches. Moreover, the PatchCore model requires only a few images for training (number of only 294 training images sampled), which is a great advantage over other unsupervised and especially supervised approaches. Beyond that, the initial very high inference time can be greatly reduced (from 2 s to 500 ms) by decreasing the size of the training dataset. Thus, both the PatchCore model and, in some use cases, the PaDiM model turn out to be highly promising candidates for vision-based anomaly detection, which demonstrates that unsupervised models are able to satisfy high performance expectations of industrial use-cases while providing sufficient flexibility to be trained without any knowledge about pre-defined defect types.

4.3. Results of the ablation studies

To verify the robustness of all unsupervised models, we also performed several ablation studies based on different configurations and hyperparameter sets. More specifically, across all models, we increased and decreased the applied image resolution and modified the split ratio between training, validation, and test data. Moreover, we examined the impact of several model-specific hyperparameters (i.e., momentum term and learning rate for Skip-GANomaly; embedding size in combination with different backbone networks for PaDiM; coreset sampling ratio and neighborhood size for PatchCore). An overview of the ablated hyperparameter values can be found in Table 9 in Appendix A.1. Beyond that, the respective authors of each model have already performed extensive ablation studies for several key parameters, which can be viewed in the respective papers (Akçay et al., 2019; Defard et al., 2020; Roth et al., 2021).

Similar to the main experiments, all configurations were performed with five different seeds. Thus, in total, we ran 200 additional experiments. To keep the computational effort manageable and conduct the experiments in a reasonable time, all runs were performed on the subset of the data collection that only contains a single VIN label layout (i.e., "rest of the world" layout). The total required time to perform the experiments was 152 h, including training and inference times.

The detailed evaluation results of all runs are reported in Appendix A.3 for each model. We summarize the aggregated results in Fig. 5. Overall, we find that the performance stays robust across different configurations and hyperparameter sets. Especially the most promising methods, PaDiM and PatchCore, show very robust results across all runs. A closer look reveals that PatchCore not only achieves the best performance, but is also characterized by a low degree of

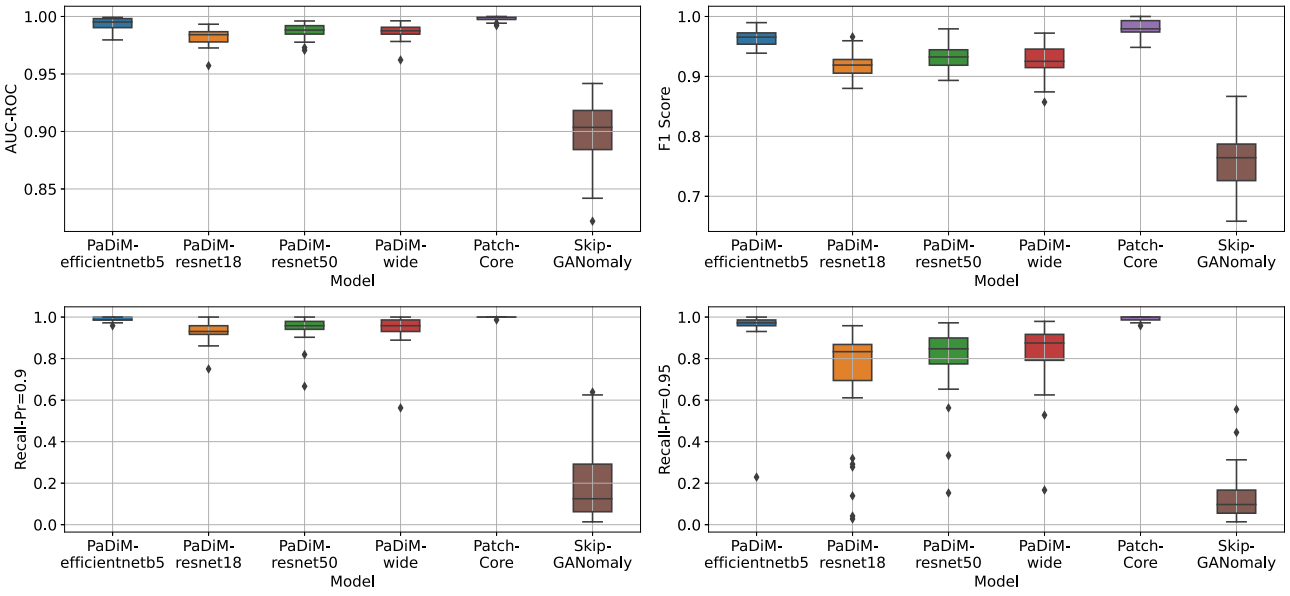

Fig. 5. Overview of the results for the ablation studies.

dispersion when varying the chosen configurations. Interestingly, we also found that by reducing the coreset sampling ratio from 25% (default value in our main experiments) to 1%, we were able to achieve even better performance results. In the ablation studies, we used a ratio of 1% as the baseline value for all ablation runs as it was intended to drastically reduce the training time. At the same time, we expected some degradation in detection performance. However, as it turned out, not only could the training time be reduced by a factor of about ten, but also the detection performance improved slightly (cf. Table 17).

The second best approach, PaDiM, shows small performance differences depending on the choice of the respective backbone network. As already noted in the main experiments, EfficientNetB5 turns out to be the best performing backbone. After that, as expected, the performance values of WideResNet50, ResNet50, and ResNet18 line up in descending order. In terms of robustness, all four variants show a similar degree of dispersion across all performance metrics, with EfficientNetB5 showing the most robust results. This is particularly evident when looking at the results for $Recall_{Pr=0.9}$.

As indicated in Fig. 5, Skip-GANomaly performs worst, having the lowest detection performance and the highest degree of dispersion across all metrics. This reconfirms that the model is not suitable for the given anomaly detection task, even when slightly modifying several hyperparameters and configurations.

5. Discussion

Deep learning is becoming increasingly important in manufacturing environments, especially for the prediction of remaining useful life of machinery (Zhang et al., 2021a) or damage detection in industrial applications (Avola et al., 2022; Olimov et al., 2022). High demands are placed on the algorithms for automated quality inspection, which are primarily aimed at maximizing *Recall* for a minimum *Precision* threshold result (Zhang et al., 2021b). For the case at hand, this means that 99.6% of the damage detected by a model should actually be damage, or else additional burdens for the production site will result. On the other hand, overlooking damage in the automotive industry may prevent faulty vehicles that have already been shipped from being accepted in the importing country and having to be transferred back to the production plant, which also results in high costs and wasted resources.

Unsupervised learning can be particularly useful in these cases of visual quality assurance as it allows the algorithm to learn from images without the need for explicit annotations or examples of defects. This can be especially useful in cases where it is difficult or impractical to manually annotate a large number of images, where the appearance of defects may vary significantly from one image to the next, or where the defect types might change over time.

Against this background, we performed an in-depth comparative evaluation of three unsupervised deep learning models for the task of industrial anomaly detection and compared it to the competitive results of a supervised model. Although the supervised counterpart achieves nearly perfect detection results by identifying almost all damaged VIN labels, its use in production environments is ruled out for several reasons. First, the defect types might not be known beforehand, and a manufacturer is generally interested in avoiding any risk by identifying all abnormal instances. This poses a problem to supervised models which depend on this prior knowledge. Second, even if the defect types are known beforehand, each error type also needs to be represented by a significant number of instances. Although specific research streams, like one-shot learning or few-shot learning are trying to address this problem (e.g., Lin et al., 2021), the majority of deep learning models still depends on large amounts of instances. Third, over time, new defect types might occur because of small changes in dynamically evolving production lines (e.g., change of resources, change of personnel, etc.). Supervised models would have to be retrained on data that specifically represents these new defect types. In contrast, unsupervised models, such as PaDiM and PatchCore do not rely on a such retraining, as they simply find abnormal instances. Against this background, we find that unsupervised models can achieve close to perfect results. Importantly, they do so without the need of any labeled instance during training. Instead they only require undamaged instances. These undamaged instances are often available in large amounts and are considerably cheaper than elaborately hand-labeled instances.

With our study, we contribute to research and practice in several ways. While the majority of previous research at the intersection of deep learning and visual anomaly detection has been primarily concerned with the development of new methods under laboratory conditions as well as comparative evaluation studies on benchmark datasets, we provide new insights on the use of unsupervised models under real-world conditions that may not be evident otherwise. Similarly, unlike broad benchmark studies, which usually examine a large variety of different models (e.g., Chadha et al., 2019; Siegel, 2020; Zheng et al., 2022), our focus was specifically on a few individual models. However, the few selected ones have been analyzed in a

Table 9
Choice of hyperparameters for model training and evaluation. The implemented default values for the main and additional experiments are highlighted by underlines.

Model	Tuning parameters	Tuning range examined by the model developers	Recommended default value	Implemented (and ablated) values in this study
Skip-GANomaly	Test split ratio	–	–	0.1, 0.15, 0.2
	Image resolution	32	32	208×208, 256×256, 304×304
	Size of the latent z vector	100, 256, 512, 1024	100	100
	Weight adversarial loss	1, 10, 20, 30, …, 80, 90	1	1
	Weight contextual loss	1, 10, 20, 30, …, 80, 90	50	50
	Weight latent loss	1, 10, 20, 30, …, 80, 90	1	1
	Number of epochs	–	15	30
	Learning rate policy	lambda, step, plateau	lambda	lambda
	Learning rate	–	0.0002	0.00002, 0.0002, 0.002
	Momentum term beta1	–	0.5	0.25, 0.5, 0.75
	Number iterations decay	–	100	100
PaDiM	Test split ratio	–	–	0.1, 0.15, 0.2
	Image resolution	–	256×256	208×208, 256×256, 304×304
	Backbone	ResNet18, WideResNet50, EfficientNetB5	WideResNet50	ResNet18, ResNet50, WideResNet50, EfficientNetB5
	Embedding size	ResNet18: $n = 100$, WideResNet50: $n = 550$	ResNet18: $n = 100$, WideResNet50: $n = 550$	ResNet18: n =448, 336, 224, ResNet50: n =550, 412, 275, WideResNet50: n =550, 412, 275, EfficientNetB5: n =344, 258, 172
PatchCore	Test split ratio	–	–	0.1, 0.15, 0.2
	Image resolution	224, 288, 360, 448	256×256	208×208, 256×256, 304×304
	Backbone	ResNet18, ResNet50, WideResNet50, EfficientNetB5	WideResNet50	WideResNet50
	Memory bank subsampling	coreset, random, learned	coreset	coreset
	Coreset sampling ratio	0.01, 0.1, 0.25	0.25	0.01, 0.1, 0.25
	Neighborhood size	1, 2, …, 6, 7	3	1, 3, 5
	Feature hierarchies	1, 2, 3, (1+2), (2+3), (1+2+3)	(2+3)	(2+3)
	Low shot ratio	1, 2, 5, 10, 16, 20, 50, 100	100	100
AutoClassifier	Test split ratio	–	0.15	0.15
	Image resolution	–	–	256×256
	Number of epochs	–	100	30
	Learning rate	–	0.001	0.001
	Momentum	–	0.9	0.9
	Backbone	ResNet18, ResNet34, ResNet50, ResNet101, VGG11, VGG16, VGG19, Fusion	–	ResNet18, ResNet34, ResNet50, ResNet101, VGG11, VGG16, VGG19, Fusion

very comprehensive and detailed manner, including several evaluation metrics, different variations of the data collection, extensive ablation studies based on different configurations and model hyperparameters, and a qualitative model assessment, which often tend to be neglected in traditional benchmark studies.

On this basis, it was possible to systematically examine the merits and limitations of each individual model. Thus, our results offer a realistic representation of the complexities and nuances of a practically relevant anomaly detection problem. This has led to findings and recommendations that are not only relevant for researchers and model developers but also for practitioners in the field who are interested in using such state-of-the-art methods for their own inspection tasks. Hence, it turned out, for example, that the representation-based methods (i.e., PaDiM and PatchCore) perform far better than the reconstruction-based approach using Skip-GANomaly. Nevertheless, across all comparisons, there was no single model that performed best in all constellations and across all metrics. Therefore, it is not feasible to focus on a single model when implementing it in real operations. Instead, it requires a situation-specific selection of an adequate model. Furthermore, it could be shown that individual pre-processing steps (e.g., layout correction in reading direction) as well as complexity constraints (e.g., restriction to single layout) could improve the performance of all models. Such concrete insights could be fed back into the superior inspection environment, where appropriate adjustments can ensure that the deep learning models deliver optimal results. For our cooperation partner, these detailed aspects are of utmost importance, because due to the strict quality standards in industrial production, it must be ensured that no inferior model is used by mistake, as this can lead to considerable costs in the event of unnoticed damage.

In order to conduct our in-depth study, an extensive database was necessary. Although the unsupervised models did not require any labeled images of damage patterns during model training, labeled data was still needed to systematically evaluate the models' performance. For this purpose, we created a comprehensive dataset containing a large number of damages. However, instead of generating synthetic examples of defects using image editing tools, we used real VIN labels and physically induced various damage types, such as cracks, bubbles, and edge defects, which were subsequently glued on surfaces of test vehicles to receive realistic examples of anomalous cases. For the car manufacturer, this data collection is a highly valuable resource, as it serves as an important reference for model evaluation. Thus, without these ground truth examples, it would not have been possible to accurately assess the models' performance and determine how well they are able to identify anomalies under real conditions. In this context, our findings are also important for other car manufacturers who may not have the necessary resources to produce such damage examples themselves. As such, our results provide important insights into which models are well suited to the task at hand without the need to re-evaluate them in another extensive model evaluation.

Overall, our results demonstrate that unsupervised models achieve highly promising results in real-world applications. This makes them not only appealing in the sector of manufacturing, but also in other cases. In the e-commerce industry, unsupervised models can be used to verify the quality and accuracy of product images and videos on online marketplaces or company websites. This can help improve the customer experience and reduce the risk of returns or customer complaints. In the advertising industry, unsupervised approaches can be used to verify the quality and accuracy of images in marketing materials, such as advertisements, brochures, or social media posts. This can help ensure

that the company's branding and messaging is consistent and effective. Finally, in the healthcare industry, unsupervised models can be used to verify the quality and accuracy of medical images, such as X-rays, MRIs, or CT scans. This can help ensure that medical professionals have access to clear and accurate images that can be used for diagnosis and treatment. In the healthcare domain, unsupervised approaches are particularly appealing since labeling must be done by expensive medical experts, the number of instances of diseases is usually low, and data sharing is oftentimes legally restricted.

As with any research, our study is not free of limitations. Thus, in order to conduct our in-depth evaluation, we had to limit ourselves to a small number of anomaly detection models. Suitable candidates were chosen based on previously established selection criteria, such as that the approach was sufficiently generic to be applicable to the case at hand, and that appropriate source code was available to facilitate implementation. However, despite the establishment of such selection criteria, several promising alternatives remained in the end that could not be included in our in-depth analysis due to resource constraints. As such, it is planned in future research to screen the results of other alternative approaches to successively integrate potential candidates into our analysis pipeline. A good starting point for this endeavor is provided by other comparative evaluation studies, such as those conducted by Zheng et al. (2022), in which recent model developments are evaluated on public benchmark datasets.

Second, due to very long training times of some models (e.g., PatchCore, Skip-GANomaly), we were also limited in performing more extensive model runs. Currently, all experiments were consistently run with five different seeds to provide a fair model comparison and evaluate the models' robustness under slightly modified conditions. However, the standard deviations for the more critical metrics $Recall_{Pr=0.996}$ and $Recall_{Pr=0.95}$ imply that the unsupervised models partly suffer from robustness, which can only be determined in a reliable manner by a higher number of model runs. Yet, since some models already require several hours of training time even for a single run (cf. Table 4), the total time of all experiments would take up to several weeks or months. As a compromise, we therefore performed several extensive ablation studies on a reduced subset of data with a single VIN layout, where the generated results also provide an indicator of how robust the results of the individual models are. Furthermore, it is important to note that the threshold-dependent metrics $Recall_{Pr=0.996}$ and $Recall_{Pr=0.95}$ represent very tight specifications, where even minimal misclassifications result in strong performance drops. By contrast, if we consider the *AUROC* values as a threshold-invariant metric, we can see that indeed all models show quite robust results.

A last limitation, concerns the examination of further hyperparameter sets for the individual unsupervised models. Thus, despite extensive experiments, including our comprehensive ablation studies, there are further hyperparameters that could be examined in more detail in future investigations. Currently, for most hyperparameters, we have implemented the default values recommended by the respective authors, some of which have already been determined to be optimal in prior ablation studies. However, as with any machine learning project, improvements may be possible by implementing different hyperparameters for the specific case at hand. To this end, our documented results serve as a solid starting point to assess which hyperparameters, that have not yet been optimized, offer promising directions for further investigation.

Finally, it is important to note that unsupervised learning is just one tool that can be used in visual quality assurance, and it may not always be the most appropriate approach depending on the specific needs and goals of the system. For example, supervised learning may be more suitable in cases where the appearance of defects is more consistent and can be more easily defined.

6. Conclusion

In this work, we showed how unsupervised learning can be used in the context of visual quality assurance to identify defects or anomalies in images. We trained a variety of unsupervised learning models on a large dataset of images showing undamaged labels of vehicle identification numbers. The models accurately learned to identify patterns and features that are characteristic of a defect-free label, and then used this knowledge to flag any deviations from these patterns as potential defects.

CRediT authorship contribution statement

Justus Zipfel: Conceptualization, Methodology, Software, Validation, Data curation, Writing – original draft, Visualization. **Felix Verworner:** Conceptualization, Methodology, Software, Data curation. **Marco Fischer:** Conceptualization, Supervision, Data curation, Project administration, Resources. **Uwe Wieland:** Supervision, Project administration, Funding acquisition, Resources. **Mathias Kraus:** Writing – original draft, Writing – review & editing, Visualization. **Patrick Zschech:** Conceptualization, Supervision, Writing – original draft, Writing – review & editing.

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 data that has been used is confidential.

Funding

This research was partially supported by the Federal Ministry of Education and Research (BMBF), Germany within the project "White-Box-AI" (grant number 01IS22080) and managed by the project management agency Deutsches Zentrum für Luft- und Raumfahrt e. V. (DLR), DLR-Projektträger.

Appendix

A.1. Choice of hyperparameters

See Table 9.

A.2. Data split for additional experiments

See Table 10.

A.3. Detailed results of the ablation studies

See Tables 11–17.

Appendix B. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.cie.2023.109045.

Table 10
Separation of the dataset into training, validation, and test split.

	Total		Training		Validation		Test	
	Undamaged	Damaged	Undamaged	Damaged	Undamaged	Damaged	Undamaged	Damaged
Supervised (all)	3673 (100%)		2571 (70%)		550 (15%)		552 (15%)	
	2703 (74%)	970 (26%)	1892 (74%)	679 (26%)	405 (74%)	145 (26%)	406 (74%)	146 (26%)
Unsupervised (all)	3673 (100%)		1892 (52%)		1229 (33%)		552 (15%)	
	2703 (74%)	970 (26%)	1892 (100%)	–	405 (33%)	824 (67%)	406 (74%)	146 (26%)
Unsupervised (stationary)	3155 (100%)		1529 (48%)		1152 (37%)		474 (15%)	
	2185 (69%)	970 (31%)	1529 (100%)	–	328 (28%)	824 (72%)	328 (69%)	146 (31%)
Unsupervised (mobile)	1488 (100%)		362 (24%)		902 (61%)		224 (15%)	
	518 (35%)	970 (65%)	362 (100%)	–	78 (9%)	824 (91%)	78 (35%)	146 (65%)
Unsupervised (single layout)	1398 (100%)		642 (46%)		546 (39%)		210 (15%)	
	918 (66%)	480 (34%)	642 (100%)	–	138 (25%)	408 (75%)	138 (66%)	72 (34%)

Table 11
Results of the main experiments for the Auto-Classifier based on different backbones.

Backbone	*AUROC*	*F1 Score*	$Recall_{Pr=0.996}$	$Recall_{Pr=0.95}$	$Recall_{Pr=0.9}$
VGG11	1.000 ± 0.000	0.988 ± 0.004	0.774 ± 0.433	0.996 ± 0.006	1.000 ± 0.000
VGG16	1.000 ± 0.000	0.994 ± 0.004	0.945 ± 0.085	1.000 ± 0.000	1.000 ± 0.000
VGG19	1.000 ± 0.000	0.992 ± 0.004	0.967 ± 0.048	0.999 ± 0.003	0.999 ± 0.003
ResNet18	1.000 ± 0.000	0.989 ± 0.011	0.962 ± 0.041	0.999 ± 0.003	0.999 ± 0.003
ResNet34	1.000 ± 0.000	0.996 ± 0.004	0.988 ± 0.017	0.999 ± 0.003	1.000 ± 0.000
ResNet50	1.000 ± 0.000	0.995 ± 0.005	0.995 ± 0.009	1.000 ± 0.000	1.000 ± 0.000
ResNet101	1.000 ± 0.000	0.995 ± 0.003	0.763 ± 0.432	1.000 ± 0.000	1.000 ± 0.000
Fusion	1.000 ± 0.001	0.995 ± 0.006	0.792 ± 0.443	1.000 ± 0.000	1.000 ± 0.000
AutoML	1.000 ± 0.000	0.992 ± 0.003	0.978 ± 0.022	1.000 ± 0.000	1.000 ± 0.000

Table 12
Results of the ablation studies for Skip-GANomaly. Baseline parameters are image resolution = 256×256, test split ratio = 0.15, learning rate = 0.0002, beta1 = 0.5.

Hyper-parameter	Value	*AUROC*	*F1 Score*	$Recall_{Pr=0.996}$	$Recall_{Pr=0.95}$	$Recall_{Pr=0.9}$
(baseline)	–	0.910 ± 0.020	0.769 ± 0.039	0.167 ± 0.077	0.172 ± 0.084	0.206 ± 0.081
Image resolution	128×128	0.905 ± 0.023	0.792 ± 0.043	0.078 ± 0.045	0.078 ± 0.045	0.147 ± 0.157
	512×512	0.883 ± 0.017	0.730 ± 0.035	0.100 ± 0.054	0.100 ± 0.054	0.125 ± 0.053
Test split ratio	0.1	0.892 ± 0.029	0.747 ± 0.039	0.096 ± 0.043	0.096 ± 0.043	0.175 ± 0.146
	0.2	0.894 ± 0.025	0.754 ± 0.036	0.092 ± 0.066	0.110 ± 0.102	0.121 ± 0.113
Learning rate	0.00002	0.899 ± 0.041	0.758 ± 0.055	0.222 ± 0.079	0.350 ± 0.135	0.414 ± 0.185
	0.002	0.900 ± 0.019	0.775 ± 0.054	0.056 ± 0.028	0.056 ± 0.028	0.061 ± 0.037
Momentum beta1	0.25	0.913 ± 0.017	0.760 ± 0.032	0.125 ± 0.026	0.125 ± 0.026	0.211 ± 0.116
	0.75	0.908 ± 0.020	0.763 ± 0.041	0.125 ± 0.055	0.125 ± 0.055	0.247 ± 0.112

Table 13
Results of the ablation studies for PaDiM (EfficientNetB5). Baseline parameters are image resolution = 256×256, test split ratio = 0.15, embedding size = 344 (100%).

Hyper-parameter	Value	*AUROC*	*F1 Score*	$Recall_{Pr=0.996}$	$Recall_{Pr=0.95}$	$Recall_{Pr=0.9}$
(baseline)	–	0.995 ± 0.003	0.970 ± 0.011	0.581 ± 0.325	0.972 ± 0.015	0.989 ± 0.010
Image resolution	208×208	0.995 ± 0.005	0.971 ± 0.012	0.678 ± 0.352	0.981 ± 0.011	0.992 ± 0.007
	304×304	0.993 ± 0.004	0.959 ± 0.013	0.664 ± 0.305	0.961 ± 0.018	0.989 ± 0.006
Test split ratio	0.1	0.992 ± 0.007	0.961 ± 0.024	0.629 ± 0.281	0.821 ± 0.297	0.988 ± 0.017
	0.2	0.995 ± 0.003	0.958 ± 0.009	0.600 ± 0.243	0.975 ± 0.017	0.992 ± 0.008
Embedding size	258 (75%)	0.993 ± 0.005	0.964 ± 0.009	0.636 ± 0.327	0.978 ± 0.014	0.992 ± 0.007
	172 (50%)	0.993 ± 0.004	0.971 ± 0.009	0.644 ± 0.307	0.981 ± 0.011	0.989 ± 0.010

Table 14
Results of the ablation studies for PaDiM (ResNet18). Baseline parameters are image resolution = 256×256, test split ratio = 0.15, embedding size = 448 (100%).

Hyper-parameter	Value	*AUROC*	*F1 Score*	$Recall_{Pr=0.996}$	$Recall_{Pr=0.95}$	$Recall_{Pr=0.9}$
(baseline)	–	0.985 ± 0.002	0.901 ± 0.021	0.519 ± 0.201	0.839 ± 0.056	0.919 ± 0.037
Image resolution	208×208	0.987 ± 0.004	0.935 ± 0.020	0.356 ± 0.272	0.869 ± 0.045	0.972 ± 0.026
	304×304	0.980 ± 0.004	0.923 ± 0.013	0.253 ± 0.172	0.553 ± 0.326	0.933 ± 0.018
Test split ratio	0.1	0.980 ± 0.012	0.919 ± 0.020	0.300 ± 0.201	0.733 ± 0.347	0.900 ± 0.076
	0.2	0.986 ± 0.006	0.915 ± 0.015	0.475 ± 0.279	0.798 ± 0.110	0.952 ± 0.028
Embedding size	336 (75%)	0.981 ± 0.005	0.919 ± 0.012	0.239 ± 0.189	0.581 ± 0.302	0.931 ± 0.032
	224 (50%)	0.979 ± 0.005	0.921 ± 0.023	0.211 ± 0.185	0.639 ± 0.200	0.917 ± 0.037

Table 15
Results of the ablation studies for PaDiM (ResNet50). Baseline parameters are image resolution = 256×256, test split ratio = 0.15, embedding size = 550 (100%).

Hyper-parameter	Value	*AUROC*	*F1 Score*	$Recall_{Pr=0.996}$	$Recall_{Pr=0.95}$	$Recall_{Pr=0.9}$
(baseline)	–	0.987 ± 0.004	0.917 ± 0.017	0.547 ± 0.221	0.842 ± 0.059	0.950 ± 0.024
Image resolution	208×208	0.991 ± 0.006	0.955 ± 0.018	0.478 ± 0.303	0.806 ± 0.240	0.981 ± 0.014
	304×304	0.984 ± 0.006	0.925 ± 0.008	0.431 ± 0.170	0.672 ± 0.262	0.956 ± 0.031
Test split ratio	0.1	0.987 ± 0.009	0.925 ± 0.020	0.617 ± 0.295	0.817 ± 0.135	0.896 ± 0.116
	0.2	0.990 ± 0.003	0.937 ± 0.014	0.552 ± 0.193	0.869 ± 0.065	0.962 ± 0.017
Embedding size	412 (75%)	0.989 ± 0.004	0.935 ± 0.007	0.514 ± 0.274	0.850 ± 0.070	0.964 ± 0.011
	275 (50%)	0.987 ± 0.005	0.928 ± 0.015	0.469 ± 0.244	0.803 ± 0.098	0.939 ± 0.062

Table 16
Results of the ablation studies for PaDiM (Wide-ResNet50). Baseline parameters are image resolution = 256×256, test split ratio = 0.15, embedding size = 550 (100%).

Hyper-parameter	Value	*AUROC*	*F1 Score*	$Recall_{Pr=0.996}$	$Recall_{Pr=0.95}$	$Recall_{Pr=0.9}$
(baseline)	–	0.987 ± 0.004	0.926 ± 0.018	0.531 ± 0.270	0.856 ± 0.061	0.961 ± 0.020
Image resolution	208×208	0.991 ± 0.004	0.954 ± 0.008	0.556 ± 0.247	0.883 ± 0.061	0.989 ± 0.010
	304×304	0.985 ± 0.004	0.924 ± 0.016	0.383 ± 0.111	0.747 ± 0.134	0.953 ± 0.034
Test split ratio	0.1	0.985 ± 0.012	0.912 ± 0.030	0.579 ± 0.223	0.758 ± 0.298	0.871 ± 0.155
	0.2	0.989 ± 0.004	0.939 ± 0.026	0.448 ± 0.223	0.877 ± 0.103	0.962 ± 0.030
Embedding size	412 (75%)	0.987 ± 0.004	0.927 ± 0.020	0.444 ± 0.189	0.869 ± 0.032	0.947 ± 0.022
	275 (50%)	0.985 ± 0.006	0.922 ± 0.032	0.419 ± 0.208	0.817 ± 0.125	0.933 ± 0.036

Table 17
Results of the ablation studies for PatchCore. Baseline parameters are image resolution = 256×256, test split ratio = 0.15, coreset sampling ratio = 0.01, neighborhood size = 3.

Hyper-parameter	Value	*AUROC*	*F1 Score*	$Recall_{Pr=0.996}$	$Recall_{Pr=0.95}$	$Recall_{Pr=0.9}$
(baseline)	–	0.998 ± 0.002	0.986 ± 0.010	0.819 ± 0.273	0.994 ± 0.007	1.000 ± 0.000
Image resolution	208×208	0.999 ± 0.002	0.979 ± 0.012	0.842 ± 0.248	0.994 ± 0.007	1.000 ± 0.000
	304×304	0.998 ± 0.003	0.982 ± 0.005	0.797 ± 0.310	0.994 ± 0.007	0.997 ± 0.006
Test split ratio	0.1	0.999 ± 0.001	0.975 ± 0.018	0.904 ± 0.090	0.988 ± 0.017	1.000 ± 0.000
	0.2	0.999 ± 0.001	0.986 ± 0.011	0.804 ± 0.208	0.996 ± 0.005	1.000 ± 0.000
Coreset sampling ratio	0.1	0.998 ± 0.002	0.985 ± 0.010	0.822 ± 0.294	0.994 ± 0.007	1.000 ± 0.000
	0.25	0.997 ± 0.001	0.974 ± 0.005	0.700 ± 0.217	0.986 ± 0.012	1.000 ± 0.000
Neighborhood size	1	0.998 ± 0.002	0.990 ± 0.007	0.808 ± 0.302	0.994 ± 0.007	1.000 ± 0.000
	5	0.998 ± 0.002	0.990 ± 0.007	0.817 ± 0.286	0.994 ± 0.007	1.000 ± 0.000

References

Akçay, S., Atapour-Abarghouei, A., & Breckon, T. P. (2019). Skip-GANomaly: Skip connected and adversarially trained encoder-decoder anomaly detection. arXiv: 1901.08954 [cs.CV].

Alizadeh, M., & Ma, J. (2021). A comparative study of series hybrid approaches to model and predict the vehicle operating states. *Computers & Industrial Engineering, 162*, Article 107770.

Avola, D., Cascio, M., Cinque, L., Fagioli, A., Foresti, G. L., Marini, M. R., & Rossi, F. (2022). Real-time deep learning method for automated detection and localization of structural defects in manufactured products. *Computers & Industrial Engineering, 172*, Article 108512.

Bao, Y., Tang, Z., Li, H., & Zhang, Y. (2019). Computer vision and deep learning–based data anomaly detection method for structural health monitoring. *Structural Health Monitoring, 18*(2), 401–421.

Bertolini, M., Mezzogori, D., Neroni, M., & Zammori, F. (2021). Machine Learning for industrial applications: A comprehensive literature review. *Expert Systems with Applications, 175*, Article 114820.

Boell, S. K., & Cecez-Kecmanovic, D. (2014). A hermeneutic approach for conducting literature reviews and literature searches. *Communications of the Association for Information Systems, 34*(12), 257–286.

Chadha, G. S., Rabbani, A., & Schwung, A. (2019). Comparison of semi-supervised deep neural networks for anomaly detection in industrial processes. In *2019 IEEE 17th international conference on industrial informatics, vol. 1* (pp. 214–219).

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys, 41*(3), 1–58.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory, 13*(1), 21–27.

Dang, L. M., Kyeong, S., Li, Y., Wang, H., Nguyen, T. N., & Moon, H. (2021). Deep learning-based sewer defect classification for highly imbalanced dataset. *Computers & Industrial Engineering, 161*, Article 107630.

Defard, T., Setkov, A., Loesch, A., & Audigier, R. (2020). PaDiM: a patch distribution modeling framework for anomaly detection and localization. arXiv:2011.08785 [cs.CV].

Esteva, A., Chou, K., Yeung, S., Naik, N., Madani, A., Mottaghi, A., Liu, Y., Topol, E., Dean, J., & Socher, R. (2021). Deep learning-enabled medical computer vision. *npj Digital Medicine*, *4*(1), 5.

Gong, D., Liu, L., Le, V., Saha, B., Mansour, M. R., Venkatesh, S., & van den Hengel, A. (2019). Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. arXiv:1904.02639 [cs.CV].

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press, http://www.deeplearningbook.org.

Harl, M., Herchenbach, M., Kruschel, S., Hambauer, N., Zschech, P., & Kraus, M. (2022). A light in the dark: Deep learning practices for industrial computer vision. In *Proceedings of the 17th international conference on wirtschaftsinformatik*.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2018). Mask R-CNN. arXiv:1703.06870 [cs.CV].

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. arXiv:1512.03385 [cs.CV].

Huang, C., Ye, F., Zhang, Y., Wang, Y.-F., & Tian, Q. (2021). ESAD: End-to-end deep semi-supervised anomaly detection. arXiv:2012.04905 [cs.LG].

Janiesch, C., Zschech, P., & Heinrich, K. (2021). Machine learning and deep learning. *Electronic Markets*, *31*(3), 685–695.

Kang, Z., Catal, C., & Tekinerdogan, B. (2020). Machine learning applications in production lines: A systematic literature review. *Computers & Industrial Engineering*, *149*, Article 106773.

Kraus, M., Feuerriegel, S., & Oztekin, A. (2020). Deep learning in business analytics and operations research: Models applications and managerial implications. *European Journal of Operational Research*, *281*(3), 628–641.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*, 436–444.

LeDell, E., & Poirier, S. (2020). H2O AutoML: Scalable automatic machine learning. In *7th ICML workshop on automated machine learning*.

Li, T.-S. (2009). Applying wavelets transform and support vector machine for copper clad laminate defects classification. *Computers & Industrial Engineering*, *56*(3), 1154–1168.

Lin, D., Cao, Y., Zhu, W., & Li, Y. (2021). Few-shot defect segmentation leveraging abundant defect-free training samples through normal background regularization and crop-and-paste operation. In *2021 IEEE international conference on multimedia and expo* (pp. 1–6).

Lopes, V., & Alexandre, L. A. (2020). Auto-classifier: A robust defect detector based on an AutoML head. arXiv:2009.01573 [cs.CV].

Olimov, B. A. u., Veluvolu, K. C., Paul, A., & Kim, J. (2022). UzADL: Anomaly detection and localization using graph Laplacian matrix-based unsupervised learning method. *Computers & Industrial Engineering*, *171*, Article 108313.

Pang, G., Shen, C., Cao, L., & Hengel, A. V. D. (2021). Deep learning for anomaly detection: A review. *ACM Computing Surveys*, *54*(2).

Perera, P., Nallapati, R., & Xiang, B. (2019). OCGAN: One-class novelty detection using GANs with constrained latent representations. arXiv:1903.08550 [cs.CV].

Reiss, T., Cohen, N., Bergman, L., & Hoshen, Y. (2021). PANDA: Adapting pretrained features for anomaly detection and segmentation. arXiv:2010.05903 [cs.CV].

Roth, K., Pemula, L., Zepeda, J., Schölkopf, B., Brox, T., & Gehler, P. (2021). Towards total recall in industrial anomaly detection. arXiv:2106.08265 [cs.CV].

Rudolph, M., Wandt, B., & Rosenhahn, B. (2020). Same same but DifferNet: Semi-supervised defect detection with normalizing flows. arXiv:2008.12577 [cs.CV].

Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S. A., Binder, A., Müller, E., & Kloft, M. (2018). Deep one-class classification. In J. Dy, & A. Krause (Eds.), *Proceedings of machine learning research*: *vol. 80*, *Proceedings of the 35th international conference on machine learning* (pp. 4393–4402). PMLR.

Sager, C., Janiesch, C., & Zschech, P. (2021). A survey of image labelling for computer vision applications. *Journal of Business Analytics*, *4*(2), 91–110.

Siegel, B. (2020). Industrial anomaly detection: A comparison of unsupervised neural network architectures. *IEEE Sensors Letters*, *4*(8), 1–4, Conference Name: IEEE Sensors Letters.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556 [cs.CV].

Tabernik, D., Šela, S., Skvarč, J., & Skočaj, D. (2019). Segmentation-based deep-learning approach for surface-defect detection. *Journal of Intelligent Manufacturing*, *31*(3), 759–776.

Tan, M., & Le, Q. V. (2020). EfficientNet: Rethinking model scaling for convolutional neural networks. arXiv:1905.11946 [cs.LG].

Tao, X., Zhang, D., Ma, W., Liu, X., & Xu, D. (2018). Automatic metallic surface defect detection and recognition with convolutional neural networks. *Applied Sciences*, *8*(9).

Van Der Maaten, L., Postma, E., & Van den Herik, J. (2009). Dimensionality reduction: a comparative review. *Journal of Machine Learning Research (JMLR)*, *10*, 66–71.

Wolf, G. W. (2011). Facility location: concepts, models, algorithms and case studies. Series: Contributions to management science. *International Journal of Geographical Information Science*, *25*(2), 331–333, eprint: https://doi.org/10.1080/13658816.2010.528422.

Yang, J., Li, S., Wang, Z., Dong, H., Wang, J., & Tang, S. (2020). Using deep learning to detect defects in manufacturing: A comprehensive survey and current challenges. *Materials*, *13*(24), 5755.

Yi, J., & Yoon, S. (2020). Patch SVDD: Patch-level SVDD for anomaly detection and segmentation. arXiv:2006.16067 [cs.CV].

Yu, J., Zheng, Y., Wang, X., Li, W., Wu, Y., Zhao, R., & Wu, L. (2021). FastFlow: Unsupervised anomaly detection and localization via 2D normalizing flows. arXiv:2111.07677 [cs.CV].

Zagoruyko, S., & Komodakis, N. (2017). Wide residual networks. arXiv:1605.07146 [cs.CV].

Zamora-Hernández, M.-A., Castro-Vargas, J. A., Azorin-Lopez, J., & Garcia-Rodriguez, J. (2021). Deep learning-based visual control assistant for assembly in Industry 4.0. *Computers in Industry*, *131*, Article 103485.

Zhang, W., Li, X., Ma, H., Luo, Z., & Li, X. (2021a). Transfer learning using deep representation regularization in remaining useful life prediction across operating conditions. *Reliability Engineering & System Safety*, *211*, Article 107556.

Zhang, W., Li, X., Ma, H., Luo, Z., & Li, X. (2021b). Universal domain adaptation in fault diagnostics with hybrid weighted deep adversarial learning. *IEEE Transactions on Industrial Informatics*, *17*(12), 7957–7967.

Zheng, Y., Wang, X., Qi, Y., Li, W., & Wu, L. (2022). Benchmarking unsupervised anomaly detection and localization. arXiv:2205.14852 [cs].

Zheng, X., Zheng, S., Kong, Y., & Chen, J. (2021). Recent advances in surface defect inspection of industrial products using deep learning techniques. *International Journal of Advanced Manufacturing Technology*, *113*(1), 35–58.

Zonnenshain, A., & Kenett, R. S. (2020). Quality 4.0—the challenging future of quality engineering. *Quality Engineering*, *32*(4), 614–626, Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/08982112.2019.1706744.

Zschech, P., Heinrich, K., Bink, R., & Neufeld, J. S. (2019). Prognostic model development with missing labels: A condition-based maintenance approach using machine learning. *Business & Information Systems Engineering*, *61*(3), 327–343.