VU VRIJE UNIVERSITEIT AMSTERDAM ProRail

Master Thesis

Detection of defective Insulated Rail Joints (IRJs) with a semi-supervised learning-based approach

Author: Martijn Wesselius (2599666)

1st supervisor: Jakub Tomczak
daily supervisor: Leah Bekkering (ProRail)
2nd reader: Rianne de Heide

A thesis submitted in fulfillment of the requirements for the Vrije Universiteit Amsterdam Master of Science degree in Business Analytics.

February 11, 2023

Acknowledgements

This thesis is written in fulfillment of the requirements for the Master of Science degree in Business Analytics at the Vrije Universiteit Amsterdam (VU). It focuses on the possibility to detect defective Insulated Rail Joints (IRJs) with a semi-supevised learning-based approach. The research for this thesis has been conducted during a six-month gradutation internship at ProRail.

First of all, I would like to thank my supervisor Leah Bekkering from ProRail for her guidance, enthusiasm and our weekly meetings during the internship. Similarly, I thank all my other direct colleagues from the ConiditionAI and ConfigurationAI teams for their advice and the great atmosphere during the weekly office days. It all made my time at ProRail very pleasant! Furthermore, I would like to express my gratitude towards Jakub Tomczak and Rianne de Heide from the VU for taking the time to be my first supervisor and second reader respectively. Lastly, I thank my girlfriend for proofreading and my parents for their support throughout my study career.

Management summary

The insulated rail joint (IRJ) is an essential component of the Dutch railway network and typically causes network disturbances in case of malfunctioning. Automatic detection of malfunctioning IRJs on video inspection data requires reliable image processing and detection algorithms. This research explores the possibility of detecting defective IRJs by using a semi-supervised learning-based approach, whereby the detection methods are trained on images of exclusively functional IRJs. This allows for detection regardless of the type of defect and solves the issue of class imbalance between functional and defective IRJs. This thesis contributes to literature by being the first to research this approach for detection of defective IRJs and rail surfaces in general.

The research includes the development of a novel dataset of over 4250 IRJ images, which are collected by a video inspection train on the Dutch railway network. The images are manually labeled according to the presence of IRJ defects as described by ProRail. The labeled dataset is used to evaluate 30 different semi-supervised approaches for the detection of defective IRJs. Each approach is a distinct combination of two image processing heuristics (rail surface crop, end post region crop), five localization methods (L2-AE, SSIM-AE, Adversarial SSIM-AE, PaDiM-R18, PaDiM-WR50) and three scoring functions (mean, standard deviation, maximum patch). A selection of these approaches are capable of creating class separation and detect defective IRJs with reasonable accuracy.

The best performing approach applies the PaDiM-WR50 model, which is a state-of-the-art detection framework introduced by [1], utilizing a pre-trained Wide ResNet-50 network. This approach also uses the standard deviation scoring function and the end post cropped imagery. It achieves a F1-score of 0.507, recall of 0.755 and precision of 0.382. In addition, it shows that the correct detections include 88% of IRJs with spark erosion and over 95% of IRJs with squats. These outcomes reveal that it is possible to detect defective IRJs regardless of the defect, but only by also incorrectly detecting many functional IRJs. The primary reason for the false detections is the sensitivity of the methods to harmless irregularities on the metal rail surface and the subtlety of most IRJ defects.

The research concludes that the semi-supervised approach shows potential, but may not yet be suitable as a standalone solution for detecting defective IRJs. For further research it is recommended to connect track circuit failure data to the defective IRJ images to discover which ones are truly malfunctioning. Furthermore, it is worth it to implement the PatchCore framework by [2], which is shown to be even better performing than the PaDiM framework.

Contents

1 Introduction

1.1 Problem

This study is carried out in collaboration with ProRail, which is the organization responsible for the railway network in the Netherlands. The company is privately owned by the Dutch government and placed under the Ministry of Infrastructure and Water Management. The Dutch railway network consists of approximately 7021 kilometers of rail tracks, 7.071 switches and 404 railway stations. This amount of infrastructure located in a relatively small an densely inhabited country as the Netherlands, makes it one of the most densely connected rail networks in the world.

ProRail has the ambition to maximize the availability of this heavily used railway network. The company must therefore efficiently monitor rail assets and keep track of their condition. Nowadays, monitoring is largely automated with various devices mounted on special inspection trains. The monitoring data is both used to discover present defects in rail assets and to predict asset failures before they occur. This allows for more efficient maintenance scheduling which subsequently minimizes the number of network disturbances.

On behalf of the Asset Management Inspection (AMI) Renewal department, this thesis will focus on the development of a solution to predict the condition of an important rail asset, namely the the Insulated Rail Joint (IRJ). The IRJ is a crucial part of the track circuit mechanism, which is widely used for train detection and signalling on the Dutch railways. Track circuit failures are often cause by damaged IRJs and typically result in alerts referred to as Falsely Occupied Track (TOBS). The IRJ condition prediction will be focused on the presence of defects in visual image data from a video inspection train.

The importance of a prediction system based on visual image data for IRJ condition is explained by various reasons. Firstly, are IRJs and track circuits very important for the safety of the railways. Therefore, failures are handled with extreme caution and play a major role in networks disturbances. Secondly, previous attempts to predict IRJ condition with other types of monitoring data have not been sufficient. It is difficult to create an all-encompassing condition statement as IRJs can suffer from various types of damages. Lastly, ProRail has received fines for not being able to sufficiently predict IRJ condition at locations such as the rail yard of Kijfhoek. An image-based prediction system is hoped to contribute significantly to an overall solution for each issue.

1.2 Research aim

The detection of defective IRJs is a typical anomaly detection task as the majority of images show undamaged IRJs. Furthermore, do defective IRJs cause track circuit failures due to multiple types of damages with each their own characteristics. Although this task can be addressed as a supervised learning problem, this thesis will aim to solve it with a semi-supervised learning-based approach. Motivation for this is three-fold. Firstly, does the semi-supervised approach solve the data imbalance problem as the associated methods only require images of functional IRJs for training. Secondly, it allows for one single method to detect malfunctioning IRJs regardless of the type, size or shape of the defects. On the contrary, supervised methods require to be trained on a balanced dataset of functional and defective IRJs and are typically optimized to detect one type of defect. This would imply the development of multiple models which is impractical. Lastly, the AMI Renewal department has never researched the semi-supervised approach for the detection of defects. Even more significantly, there is also an absence of literature on this specific topic. To our knowledge,

this study is unique as it will be the first to apply semi-supervised learning methods for the detection of defective IRJs and rail defects in general.

The aim of this research is in essence to examine the possibility to detect defective IRJs with a semi-supervised learning-based approach. It will investigate various methods which do not require prior knowledge of the various IRJ defects to be appled for detection. Their performance will be measured on the basis of a manually labeled dataset of functional and defective IRJs. Furthermore, the impact of image cropping on the prediction performance will be examined. During this research, the following research question will be addressed:

How and to what extent can a semi-supervised learning-based approach be applied to detect IRJs with various defects?

This is answered on the basis of the following sub-questions:

1. ***How to express IRJ abnormality with semi-supervised learning methods?***
2. ***Which detection approach is most promising in terms of test set performance?***
3. ***Which IRJ image crop is the most promising in terms of test set performance?***

1.3 Outline

The remaining part of this work is structured as follows:

- **Chapter 2** discusses the background concerning railway inspection, track circuits, IRJs and associated defects.
- **Chapter 3** introduces anomaly detection and discusses related work, which covers literature on rail surface defect detection and general semi-supervised defect detection for image data.
- **Chapter 4** examines the IRJ images from the video inspection train and discusses the procedures for labelling, cropping and splitting of the dataset.
- **Chapter 5** states our semi-supervised learning-based approach and explains the implemented methodology. Furthermore, it discussed the evaluation metrics to measure performance.
- **Chapter 6** states the parameters of the methodology and the procedures for training, calibration and testing. Subsequently, it presents the results of these procedures.
- **Chapter 7** discusses the results and highlights limitations. Furthermore, it concludes by answering the research questions and by providing recommendations.

2 Railway inspection and IRJs

This chapter explains how ProRail performs railway inspection and introduces train detection systems and the importance of IRJs. Furthermore, it describes the defects seen on IRJs and the surrounding rail surface.

2.1 Railway inspection

ProRail is responsible for the management, maintenance and expansion of the Dutch railway network, which consists of approximately 7021 kilometers of rail tracks, 7.071 switches and 404 railway stations. Their ambition is to optimize the networks availability and facilitate the development of the infrastructure. Despite the heavy railway usage each day, the organization is able to keep the network punctual and reliable. The actual maintenance and construction of the railway is however not performed by ProRail itself, but outsourced to certified railway contractors and other specialized maintenance companies. This happens on the basis of Performance Oriented Maintenance contracts (PGOs), which are performance agreements between ProRail and the contractor to stimulate innovation. Each PGO is linked to one of 21 maintenance regions in the Netherlands and re-assigned every 5 years. At the time of writing, the main contractors are [REDACTED].

In order to guarantee the high standards expected from the railway network and the contractors, ProRail inspects all its infrastructure at least once a year. The following types of data are then collected and evaluated:

- **Configuration data**: describes the location of rail parts and their usage measured in number of train passages or amount of axle load.
- **Condition data**: describes the current condition of rail parts as obtained from measurements and inspection.

ProRail stores and analyzes most of this data in their Sector-Wide Monitoring System (BBMS). The monitoring system establishes an assessment of the network's condition and notifies when certain components are in need of replacement. Maintenance contractors also have access to BBMS and use the information to adjust their maintenance. This ensures that components do less often fail unexpectedly and cause disturbances of train services.

In the past, condition data was manually collected by patrols walking along the rail track. To improve safety and reduce train schedule interruptions, ProRail largely replaced manual patrols by more automated data collection techniques. Apart from being safer, it is more accurate and enables the collection of a larger variety of condition data. According to [3], the existing inspection technologies include:

- **Ultrasonic inspection**: reflections of ultrasonic beams help to detect internal rail defects.
- **Eddy current inspection**: disturbances of a magnetic field help to detect rail surface and near-surface defects.
- **Acoustic emission inspection**: inspection based on the noise produced in the wheel-rail interface of the moving inspection train.
- **Visual video inspection**: inspection based on video recordings with line scan cameras to detect external rail defects, as can be seen in Figure 2.

Figure 2: The video inspection train equipped with line scan camera equipment.

This research will focus on the image data captured during visual video inspection. Since 2008, ProRail has increasingly invested in the usage this type of inspection for both configuration and condition data. Back than and still today, inspection of the video recordings is largely done by hand. Due to the enormous amount of imagery required to capture the entire railway network, manual inspection is very time consuming, tedious and also error prone. Fortunately, computer vision algorithms are able to take over this manual inspection. Algorithms are faster, cheaper and sometimes more accurate than their human counterparts.

Starting in 2018, ProRail began implementing computer vision algorithms with a focus on supplementing the configuration data. The ConfigurationAI team of the AMI Renewal department developed algorithms for detecting IRJs, recognizing rail dampener types, and identifying rail sleeper types during this time. In 2021, the focus expanded to the development of algorithms for the extraction of conditional information as well. This became the responsibility of the newly formed ConditionAI team. Their initial project involved developing a segmentation algorithm for railway fasteners, which are assets used to secure the rail head to the rail sleepers. The model can automatically identify the shape of a fastener and reject it if it's abnormal. In the future, ProRail likes to add more condition predictions based on image data to BBMS to improve maintenance planning. This thesis contributes to that goal.

2.2 Train detection systems

The Dutch railway infrastructure consists of many individual rail assets which all have to perform optimally for the network to function and remain safe. In an ideal scenario, ProRail would be up to date about the condition of each individual part at any time. However, this is not feasible at the moment, which is why ProRail prioritizes some components over others when monitoring condition.

2.2.1 Track circuit and IRJs

One component of high priority is the Insulated Rail Joint (IRJ), which is important for the functioning of the most prevalent train detection system in the Dutch railways, the track circuit mechanism. This is an electric system working on an insulated segment of rail track. This segment, which can have a length ranging from several meters to kilometers, is used to transfer current from a power source to a relay on the other end. As long as the segment is not occupied, the relay is being energized and for instance turns on the green signal light, notifying approach trains that the segment is free. When a train enters the section, its wheels and axles short the electrical circuit, preventing the current from reaching the relay. This de-energizes the relay and activates a red signal light, warning approaching trains to stay clear of the track for safety. A schematic overview of a track circuit is shown in Figure 3.

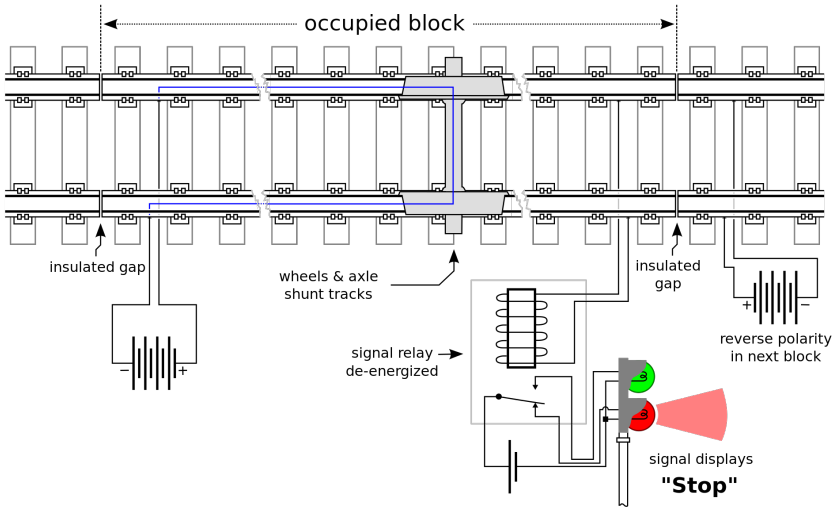


Figure 3: Occupied track circuit

The IRJ plays a crucial role in ensuring that each track circuit segment is electrically insulated and that the current cannot escape into another segment. This special type of joint is depicted in Figure 4. It separates adjacent track circuits by using a 6mm gap filled with a high strength composite plate known as an end post. The two rail ends are secured together with two insulated joint bars on each side.

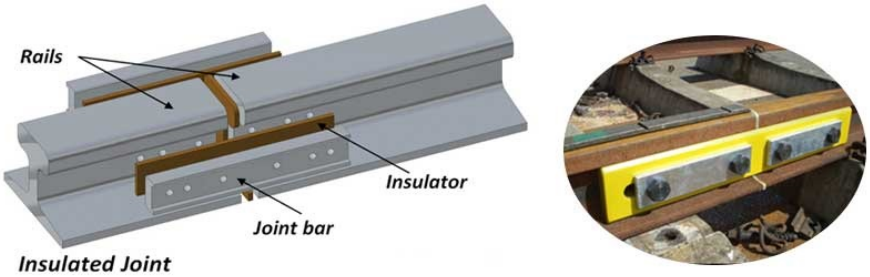


Figure 4: Insulated rail joint

The principle of short-circuiting makes the track circuit fail-safe, meaning that the system will always indicate occupation of the segment when one of its components fails. Such an event is known as a Falsely Occupied Track (TOBS) and extremely important for railway safety, but it also causes disturbances and limits network availability. ProRail found that track circuits failures are the third most occurring cause of network disturbances and are roughly 90% of the time resulting from damages to IRJs. All in all, defective IRJs account for about 15% of network disturbances and therefore it is essential to monitor and predict the condition of these assets. The actual damages responsible for track circuit failures are addressed in Section 2.3.

2.2.2 Other

The track circuit is the most common train detection system in the Dutch railways and will still be used for decades to come. However, it not the only type of detection system as two replacing systems exist.

Axle counters. Axle counters are sensors mounted near the rail tracks and count the number of passing axles to determine the presence of a train. An axle counter system includes a sensor installed at either end of a rail track segment and an evaluator which compares the counts of both sensors. The segment is occupied at the moment a train passes the first sensor. Only when the second sensor has counted the same amount of axles as the first, the segment is cleared. Axle counters are more reliable than track circuits and used as a replacement.

ECTS. The European Train Control System (ECTS) is a collection of communicating train and track-side devices and part of a European system of standards for signalling of railways (ERTMS). It mainly promotes the interoperability of trains in Europe, enables higher train speeds and also improves safety. Different to track circuits and axle counters, the detection with ECTS is partly done with equipment onboard of the train. This onboard equipment receives the location of the train when it passes fixed transponders on rail sleepers, called Eurobalises. The onboard device calculates the live train location with the received location and the travelled distance since passing. Therefore, ECTS is the most precise train detection system to date. Over the coming 30 years, the introduction of ECTS and ERTMS will eventually lead to the disappearance of other train detection systems. At the time of writing, the route from Amsterdam to Utrecht is the only Dutch rail route using these new systems.

2.3 Condition assessment

The Dutch rail network contains around 44,000 IRJs of which between 3,000 and 3,500 are replaced each year. IRJs have an average lifespan of 10 to 15 years and are replaced preventively during renewal projects or because of excessive wear and malfunctioning. ProRail has specified various condition criteria for IRJs and defects of surrounding rail surface, which are linked to track circuit failures.

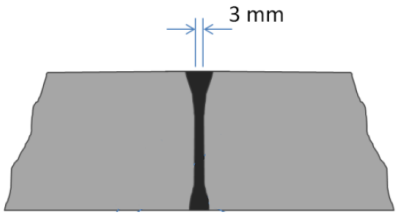
2.3.1 Defects of IRJs

Condition assessment starts with criteria specific to the construction of the IRJ. If these criteria are not met, the IRJ should be replaced:

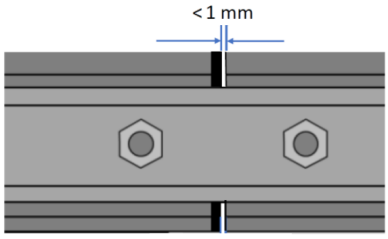
- **Spacing**: The minimum remaining thickness of the end post on the rail head surface should be at least 3mm. The original thickness of the end post is 6mm but decreases over time as trains smear the rail ends over this composite plate. It can potentially lead to a connection between two adjacent track circuits and result in TOBSs. See Figure 5a.
- **Play**: The maximum gap between the end post and either one of the adjacent rail heads has to be less than 1mm. These gaps can be a result of rail contraction caused by temperature fluctuations. See Figure 5b.
- **Indentation**: The maximum vertical indentation of the rail track surface around the end post should be less than 0.5mm. Indentation can be caused by bouncing movement of passing trains when the end post is not completely flush with the track surface. It

can also be caused by insufficient support from the underlying rail way sleeper and ballast. See Figure 5c.

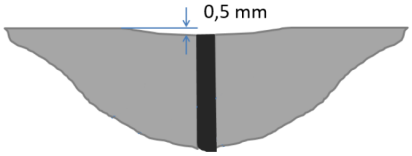
- **End post deterioration**: The end post of an IRJ should be undamaged, in the sense that is not fractured or chipped. Otherwise it could result in track circuit failures as it does not guarantee complete insulation of two adjacent segments.



(a) Minimum end post thickness



(b) Maximum end post play



(c) Maximum indentation

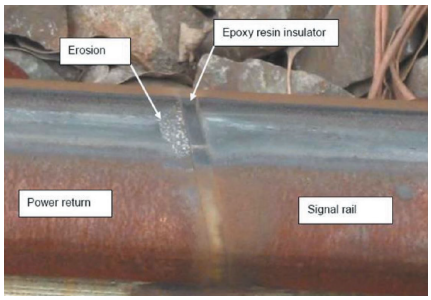
Figure 5: Defects of IRJs

2.3.2 Defects of surrounding rail surface

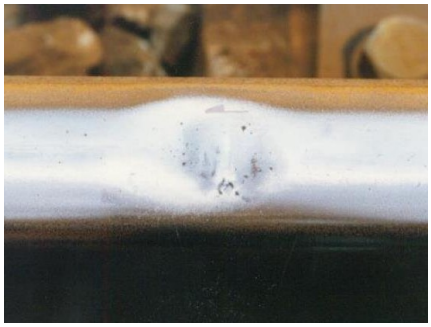
IRJ condition also considers general rail defects as IRJs are rail components. Hereby it is important to make a distinction between internal and external rail defects, as only external rail defects are visible on the video inspection data. As Chapter 4 will discuss, this study uses top-down imagery, which makes the following external rail surface defects relevant:

- **Spark erosion** is a rail surface defect which typically occurs around the IRJ end post. The intense friction of slipping train wheels and the transition from one track circuit into another, can cause an electrical arc which burns of a tiny bit of rail surface. See Figure 6a.
- **Squats** are indentations or short regions of flattened rail and can be accompanied by internal cracks in the rail. In severe cases, squats can cause crumbling or breakage of the rail. See Figure 6b.
- **Studs** are areas where the metal rail surface is flaking off, which is also caused by the high stresses created by slipping wheels.
- **Head checks** are transverse cracks on the inside of the rail surface. In severe cases, it can lead to chipping of the rail head. See Figure 6c.

- **Corrugation** is a cyclic wave-like irregularity on the rail surface. See Figure 6d.



(a) Spark erosion



(b) Squat

(c) Head check

(d) Corrugation

Figure 6: Defects of surrounding rail surface

3 Related work

The identification of defective IRJs is visual anomaly detection task. This chapter will outline this subject and identify its challenges. Additionally, it reviews previous studies on railway defect detection and underscores the requirement for a semi-supervised learning-based approach, which will be explored in greater detail by the final section.

3.1 Anomaly detection

Anomaly detection, also known as outlier or novelty detection, is generally defined as the identification of events or observations which differ from the majority of the data or from an expected normal behaviour [4]. Anomaly detection has a wide variety of applications such as credit card fraud detection, computer network intrusion detection or tumor detection on medical images. In our case, the task is to detect IRJs with end post defects or defects of the surrounding rails. Therefore, a region has to be defined which describes the appearance of normal functional IRJs. Any IRJ which strongly deviates from this region is declared an anomaly. Defect detection can be seen as an subcategory of anomaly detection as it is more specified what has to be detected. Typically defect detection is harder as non-defective and defective samples are largely similar.

Challenges. In essence, the visual anomaly detection task seems simple. However, as [4] discuss, there are several reasons why this is often challenging:

- **Imperfect boundary**: The region of normal behaviour is difficult to define since the boundary between normal and anomalous behaviour is often not precise. Normal data samples lying close to the defined boundary could actually be anomalies or vice versa. Especially with defect detection this is emphasized as normal samples and defective samples are typically very similar.
- **Noise**: The normal data samples often contain noise with similar properties as actual anomalies, making it difficult to distinguish or remove.
- **Data availability**: The availability of labeled data for training and testing of detection techniques is usually an issue. Anomalies typically occur in small quantities, making datasets of normal and abnormal samples imbalanced. Moreover, it is not guaranteed that all types of anomalies are known beforehand as anomalous behaviour is often dynamic and new anomalies might arise.
- **Varying definition**: The exact definition of an anomaly is different for each application domain. A technique developed for one domain should often be adapted for use in another domain.
- **Data dimensionality**: Specific to visual anomaly detection is the typical large dimensionality of the input data. Each image is a spatial structure of pixels with continuous attributes such as color, lightness and texture.

Modes. The mode of an anomaly detection technique describes its dependency on the availability of labelled data. The following three modes exist:

- **Supervised** anomaly detection requires labelled data from both normal and anomalous classes to train a model and optimize a boundary between normal and abnormal samples. It is the least applied mode of anomaly detection, because data imbalance and accurate labelling of defects are often encountered problems. Without these problems, supervised techniques can typically generate accurate results.

- **Semi-supervised** anomaly detection is more frequently used because it only requires partly labeled data. In most cases, the training data consists of exclusively normal samples and is used to model normal behaviour. The distance between this normal behaviour and anomalies is afterwards used to manually set a boundary.
- **Unsupervised** anomaly detection is also widely applied as it does not require any labelled data. These techniques find a boundary under the assumption that normal samples are both more frequently seen and strongly deviating from abnormal samples. When these assumptions are however not true, unsupervised methods create a high number of false positives. Unsupervised anomaly detection is typically not applicable for computer vision tasks as the high dimensionality of images makes boundaries very complex.

Detection, localization or segmentation? Images are spatial structures in which defects are typically specific region of pixels. Detection techniques for images are therefore often extensions of localization or segmentation techniques. In literature these three topics are frequently discussed together and intermixed:

- **Detection** is the task of classifying an image as either normal or anomalous.
- **Localization** is the task of assigning each pixel or region of pixels in an image with an anomaly score. The result is an localization map showing how abnormal each given pixel or region of the image is. The map can be formed into an overall anomaly score by aggregating over the pixels or into a segmentation map by setting an pixel threshold.
- **Segmentation** is the task of classifying each pixel or patch of pixels in an image as either normal or anomalous. This pixel-level classification can directly be extracted from a localization map by setting a certain threshold on the pixel values.

Localization and segmentation are harder tasks than detection. Techniques can serve an own goal or be used to explain the results of detection techniques.

3.2 Rail surface defect detection

Various researches have developed visual defect detection techniques for application in a railway setting. The topic of railway surface defect detection comes with some additional challenges as described by [5] and [6]:

1. **Illumination inequality**: Rail images are captured under a train in an open circumstance. Natural light and vibration of the train have influence on the illumination of the images.
2. **Variation of reflection property of rail surface**: The rail head does not share the same reflection characteristics over its running surface because of the curved geometry. The center of the rail surface is typically smooth and reflects more light than periphery areas which are often covered with rust.
3. **Limited features available for recognition**: Discrete rail surface defects do not share common texture or shape resulting in limited feature availability. This is mainly problematic for localization and segmentation techniques.

3.2.1 Classical methods

A large portion of rail surface defect detection models are based on image processing techniques and additionally conventional machine learning methods. These models detect and segment rail surface defects commonly in a four stage approach. First, the rail has to be extracted from the raw input image. [5] and [6] both show that the rail profile can be visualised in a histogram by projecting the average gray-scale pixel intensities over the longitudinal axis of the rail. This histogram effectively shows which columns of the image show the edges of the rail and thus how the image should be cropped to extract the rail.

Second, the illumination inequality and reflections are removed from the extracted rail image. Both [5] and [7] propose a local Michelson-like contrast measure which is nonlinear and illumination independent. For the same task, [6] implement longitudinal gray-scale equalization and [8] implement dynamic Gamma correction.

Third, the equalized rail surface is enhanced with spectral methods to make defects more pronounced. [9] find that Gabor filters work better than wavelet transforms or combined Gabor wavelet transforms. [6] implements curvature filters to remove noise an make defects easier to detect. [8] apply the most complete approach by enhancing images on sub-image level, region level and pixel level with their coarse-to-fine model (CTFM).

Fourth, the enhanced rail surface is transformed into a segmentation. This can be done manually [7], with OTSU thresholding [8] or with the proportion emphasized maximum entropy (PEME) algorithm proposed by [5], which maximizes entropy while remembering the low proportion of defects. Lastly, [6] segment defects with an enhanced Gaussian mixture model (GMM), which incorporates spatial information between neighbouring pixels.

3.2.2 Deep learning methods

More recently, focus shifted to rail defect detection techniques based on convolutional neural networks (CNNs), which are not reliant on carefully engineered image processing techniques to accurately detect defects on unevenly illuminated rail images.

Deep learning methods are far better than image processing and conventional machine learning methods in their ability to process raw data as they automatically extract multiple levels of representations. These representations are obtained by transforming input in one layer with non-linear modules into a more abstract lower layer. Enough of these layers and very abstract functions can be learned, making them very flexible in addressing high-dimensional problems, without prior assumptions about the properties of the data [10].

Especially the introduction of the convolutional layer, to which the CNN owes its name, made deep learning applicable for computer vision tasks like visual defect detection [11]. The convolutional layer makes networks more efficient to train and generalize better than networks with fully connected layers. Units in such a layer are organized in feature maps, within which each unit is connected to patches in the feature maps of the previous layer, as can be seen in Figure 7. Different feature maps uses different set of weights, but the weights within a single feature map are shared. The design is influenced by the fact that groups of values within an image are often highly correlated and local statistics are not bound to specific image regions [10]. Enough convolutional layers, in combination with non-linear activations and max pooling, make the use of image processing techniques for feature extraction redundant.

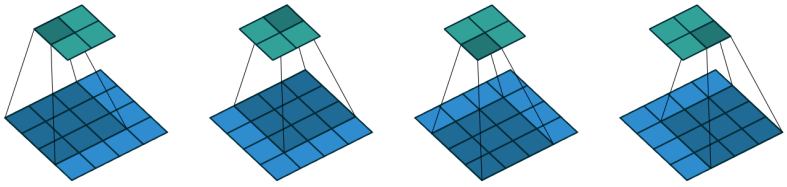


Figure 7: The convolutional layer maps each patch of features in a network layer (blue) to a single feature in the next layer (green).

[12] implement a relatively shallow CNN architecture which includes three convolutional layers, three max pooling layers followed by three fully connected layers. The authors labeled more than 20.000 images of normal rails, joints and squats in varying severity and trained the network in a supervised manner to classify each image correctly. The network is able to automatically extract features from the raw data. An interesting observation by the authors is that the last convolutional layer of their model produces output similar to what Gabor filters would do. [13] propose the use of the SegNet architecture to segment defects in images of rail surfaces. SegNet is a CNN architecture with 59 layers up- and down-sampling convolutional layers. Their implementation is trained on 120 images and shows very good results. However, testing was performed on only 13 images, which were also accurately segmented with a simple threshold. [14] implement multiple deep CNN architectures for object detection of rail surface defects. The feature extraction is performed with the well-known MobileNetV2 and MobileNetV3 CNNs. The design of detection layers with multi-scale feature maps is inspired by YOLOv3. The experimental results show that the proposed models can detect and locate the rail surface defects in real time and achieve high detection accuracy.

However, all these CNN architectures are trained in a supervised learning setting. As discussed in Section 3.1, this is actually undesirable for defect detection because defect data is often imbalanced and incomplete. Interestingly, [12] make a final remark about this in their research: *"In the current context, detection and classification of all types of rail defects is often carried out while spending much time and cost. With this in mind, exploring a deep learning approach that would be general enough to be used for automatic detection of other types of rail defects is our immediate future work of interest. In particular, we will explore the use of auto-encoders and other deep networks for this purpose"*. They refer to a semi-supervised deep learning approach, which exclusively uses undamaged rail surfaces to model normality and detect everything that deviates from this. To our knowledge, this approach has never been applied for rail defect detection.

3.3 Semi-supervised defect detection with deep learning

This section discusses the topic of semi-supervised defect detection with deep convolutional architectures. The techniques are specifically designed for image data and often applicable for defect localization as well. As mentioned, these techniques have never been applied to rail surface defects in literature.

Historically, most proposed semi-supervised techniques in literature are applied on non-defect image classification datasets such as MNIST [15] and CIFAR10 [16] of which one arbitrary image class is than relabelled as the anomalous class. The use of these datasets is motivated by a previously long lasting absence of publicly available datasets with labelled industrial defects. However, as the classes within MNIST and CIFAR10 are structurally very

different, it is unclear if the developed methods generalize well to defect detection. Fortunately, the first public multi-object and multi-defect dataset with real world applications was released in 2019: the MVTec AD dataset by [17]. This dataset contains over 5000 images of different objects and textured surfaces with more than 70 different types of defects, ranging from scratches to dents to contamination. An example of the classes is shown in Figure 8. Especially the textured surface classes in the data are comparable to rail surface defect data.

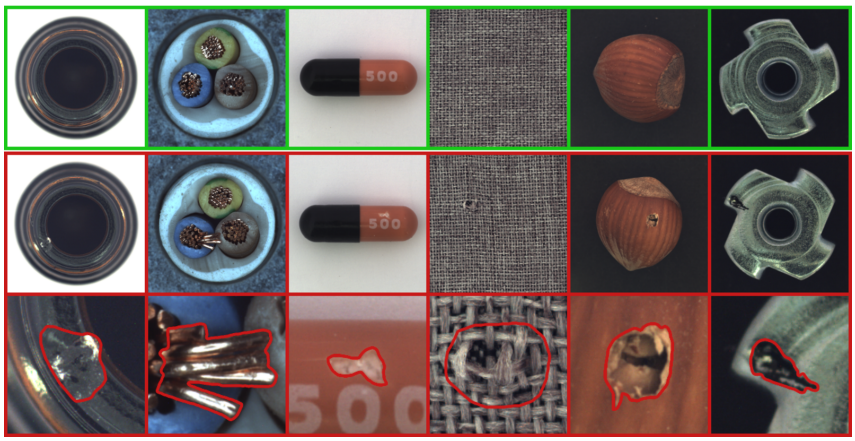


Figure 8: The image categories in the MVTec AD dataset contain normal samples (top), defective samples (middle) and segmentations (bottom).

The semi-supervised methods, which will be discussed, fall into two distinct categories. The first category covers generative models which are trained from scratch and detect defects on the basis of image reconstruction. The second category represents models which detect defects based on the feature representations extracted by pre-trained classification networks.

3.3.1 Reconstruction-based methods

The most intuitive semi-supervised approach for visual anomaly detection and localization is based on image reconstruction. This is typically performed with unsupervised learning models, such as autoencoders and generative adversarial networks, which are trained from scratch to reconstruct only normal image samples. The network learns the explicit underlying distribution of normal data as its weights are adapted to only output reconstructions which are similar to the normal image samples. The assumptions is that such network will afterwards struggle to reconstruct anomalous samples, specifically the image area deviating from normality. The reconstruction error of normal images will be small, while that of abnormal images will be large. The aggregate reconstruction error indicates the abnormality of the entire image and can be applied for detection. The pixel-wise reconstruction errors between will result in a residual map showing the location of abnormal areas in the image and is applied for localization.

Autoencoders. The Autoencoder (AE) architecture consists of an encoder part and a decoder part and is an unsupervised method to learn the identity mapping of an input. The encoder encodes the input to a latent space representation with a reduced number of dimensions, which will only retain non-redundant information. Therefore, the latent space can be seen as an approximation of the underlying distribution of the training data. After encoding, the decoder attempts to build a representation of the original input from this latent space. Traditionally, it was used for feature learning and dimensionality reduction [18], but [19]

were the first to apply it for anomaly detection on high-dimensional data. Nowadays, the Convolutional Autoencoder (CAE), with convolutions in the encoder and transposed convolutions in the decoder, is commonly used as a benchmark for visual anomaly detection [17].

Research tried to complement the CAE architecture in various ways. [20] introduced MemAE as a solution to the observation that AE are sometimes able to reproduce anomalous samples as well. The MemAE uses a memory module to save the latent space representations of normal training samples. Given an input during inference, it selects the normal latent representation which is most similar to the encoding. [21] apply multiple CAEs at different Gaussian pyramid levels while adding salt and pepper noise to the input during training to improve robustness. The reconstruction residual maps at each level are afterwards synthesised to form a final anomaly localization map.

Variational autoencoders. The Variational Autoencoder (VAE) is an adaptation of the AE enabling control over the latent space. The VAE encodes input to a latent space which is represented as a distribution instead of a point. The decoder samples a random point from this distribution to recreate an image similar to the input. The VAE is considered as a probabilistic generative model as the point sampled from the latent space influences the generated output. [22] and [23] applied VAEs for visual anomaly detection, but both do not report significant improvements over using standard AEs.

Generative adversarial networks. The Generative Adversarial Network (GAN) is a neural network architecture introduced by [24]. The network is comprised of two separate neural networks, namely a generator and a discriminator. As originally proposed, the generator attempts to generate samples from a randomly initialized latent space distribution, similar to the decoder of the VAE. The idea is that the generator tries to create realistic data while the discriminator has to distinguish what data samples are real and fake. During training they improve by penalizing each other in a zero-sum game, where one's loss is the others gain. This adversarial training component allows the GAN to capture a more accurate representation of the distribution underlying the input data and to generate more realistic samples than a VAE. One disadvantage of the GANs is the risk of mode collapse. This is when the model gets stuck in a local optimum during the early phase of training. The generator has than found a realistic output which it keeps regenerating as the discriminator is not able to distinguish from the real samples yet.

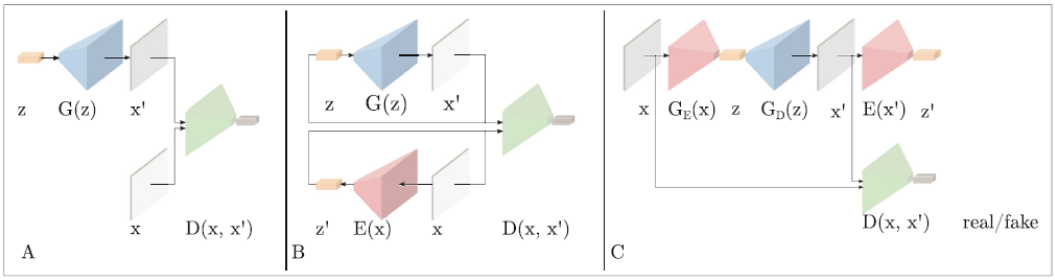


Figure 9: Architectures of AnoGAN (A), BiGAN (B) and GANomaly (C)

Researches have implemented GANs or applied the adversarial training component for visual anomaly detection in various forms. One of the first to use a Convolutional GAN (CGAN) for this cause were [25] with their AnoGAN model. Training happens in two stage. First, the generator learns to map the latent space distribution to the normal image sam-

ples and the discriminator learns to distinguish the samples from real samples. Afterwards, they make an inverse mapping of the generator, such that given a new image, a point in latent space can be selected for generation of a similar image. During inference, images are scored with a combined generator and discriminator loss. The authors show a significant improvement over a standard CAE. Later, [26] investigate the use of BiGAN model, which simultaneously learns an encoder to map input samples to latent space, along with a generator and discriminator. This avoids the computationally expensive inverse mapping step of AnoGAN. BiGAN performs slightly better than AnoGAN, but is mostly much more efficient. Then, [27] propose GANomaly, an adversarially trained encoder-decoder-encoder architecture. GANomaly jointly learns representations for both image and latent space with adversarial, contextual and encoder losses. During testing, scoring happens by the difference between the the encoding of the input sample and the encoding of the generated output. Overall, the model performs better than AnoGAN and BiGAN, while also showing faster inference times.

Loss functions. Reconstruction models are commonly trained with a per-pixel loss functions to measure the error between input and reconstruction image in terms of independent pixel brightness. L2-loss is most frequently used and implements the mean squared error (MSE) [17] as a loss function. Also L1-loss, implementing mean absolute error (MAE), is used in some cases and can lead to less blurry image reconstruction as shown by [28]. Both loss functions however share the incorrect assumption that nearby pixels or image regions are completely independent of each other. As a solution for image comparison, [29] propose Structural Similarity (SSIM), which is a perceptual difference measure more closely aligned with human perception of image quality. Unlike the per-pixel loss functions, SSIM takes into account differences in local contrast en structure, instead of only brightness. Both [30] and [23] adapt SSIM for application as a loss function for model training and conclude that it improves reconstruction and segmentation ability over per-pixel loss functions. The authors do nevertheless point out that SSIM is a parametric function and only applicable on gray-scale images.

Skip connections. Based on GANomaly, [31] propose the Skip-GANomaly model, which adapted with skip-connections similar to U-Net [32]. Skip-connections allow down-sampling layers of the encoder to be concatenated with the corresponding up-sampling layers of the decoder. The use of skip-connections provides the advantage of direct information transfer between the layers. This preserves local information of initial layers and global information of later layers and therefore yields sharper reconstructions. One shown disadvantage is however that the Skip-GANomaly model sometimes reconstruct anomalies while being trained on normal samples only. Nevertheless, detection performance is improved over the standard GANomaly architecture. Skip-connections are also applied by [33] to improve detection and localization performance of a standard CAE. Unlike U-Net, the skip-connections perform addition of layers instead of concatenation. Furthermore, the authors propose to randomly add staining noise to images during training. It shows that the CAE with only skip-connections performs worse than the standard CAE as anomalies are unintentionally reconstructed. This behaviour is however eliminated by adding noise to training images as the skip-connected denoising CAE is the best performer in their study.

3.3.2 Feature representation-based methods

The second semi-supervised anomaly detection and localization approach employs handcrafted modeling techniques on top of the feature representations extracted from normal images. The extraction of these representations, also known as embeddings, typically happens with CNNs which are trained for classification of natural images from ImageNet [34].

Interestingly, these pre-trained networks are able to extract meaningful representations from entirely different images as well and can thus be used to model normal behaviour. This section discusses four methods which use this approach and currently belong to the state-of-the-art, namely MahalanobisAD [35], PaDiM [1], SPADE [36] and PatchCore [2].

Multivariate Gaussian. [35] propose MahalnobisAD as a framework for defect detection exclusively. The techniques models normality by fitting a multivariate Gaussian (MVG) distribution on the feature representations of normal image samples. These features are extracted with an EfficientNet [37], which is trained on ImageNet. Their technique subsequently uses Mahalanobis distance between the embedding of every network layer and the modeled MVG to score the abnormality of test samples. Furthermore, by applying Principal Component Analysis (PCA) the authors find that the feature components containing little variance in normal data are the ones crucial for discriminating normal and anomalous samples. Only using the high variance components actually reduces performance.

[1] propose the Patch Distribution Modeling (PaDiM) approach on Resnet-18 [38] and Wide-ResNet-50. Broadly speaking, PaDiM is similar to MahalanobisAD as it employs MVG distribution to describe the normal feature representations and scores samples with the Mahalanobis distance. However, PaDiM applies these procedures on individual image patches instead of the entire image, making it applicable for defect localization as well. During training, the feature representations of each image patch are modeled as MVG distributions. During testing, each patch of a given sample is scored individually with the Mahalanobis distance between its feature representation and the learned distribution at that patch location. All scored patches form a localization map of which the maximum patch value acts as the anomaly score of the entire image. As the generated patch representations may carry redundant information, the authors furthermore apply dimensionality reduction before concatenation to improve inference speed. It turns out that random dimensionality reduction outperforms PCA while being more efficient. Overall, PaDiM shows an excellent combination of detection performance and inference speed.

Memory bank and K-Nearest Neighbours. The Sub-Image Pyramid Anomaly Detection (SPADE) model by [36] performs both detection and localization with feature representation of a pre-trained Wide-ResNet-50 [39]. SPADE extracts and stores the feature representations of the normal training samples at the last layer of this network. Given the representation of an image sample during testing, it runs K-Nearest Neighbor (KNN) algorithm to retrieve the K closest normal image representations from the training set. Abnormality is scored by the average distance between the test sample representation and the found K neares representations. Pixel-wise abnormality for localization is computed with the similar multi-image approach and matching of pixel feature pyramids. SPADE is a well performing method for detection and localization, but its usefulness is greatly affected by the KNN algorithm, which inference complexity scales linearly with the size of the training data.

One of the most recent additions to the feature represenation-based approach is PatchCore by [2], which uses specific components of both SPADE and PaDiM. Similar to PaDiM, PatchCore divides images into patches of which feature representations are extracted from a Wide-ResNet-50 network. The idea of PatchCore is that if a single patch is anomalous the whole image can be classified as anomalous. During training, normal patch representations are added to a memory bank, similar to SPADE. The complexity of the subsequent KNN is however reduced by coreset-subsampling of the memory bank, which approximates its structure while reducing its size. During testing, an image is scored by the maximum distance between its patch representations and the nearest neighbor normal patch representations in the memory bank. The localization map is created by realigning the patch anomaly scores

to their respective spatial location. In summary, PatchCore maximizes nominal information at test time, reduces the bias towards ImageNet and retains high inference speeds.

3.3.3 Comparison on MVTecAD

The performance on the MVTec AD dataset of several of the discussed methods can be found in the research by [33], [1] and [2]. This data is shown in Tables 1 and 2. It is observable that within the class of reconstruction models, SSIM-loss and the combined skip-connection and staining solution improves performance over a standard CAE. The GANomaly model with its adversarial training architecture also shows a slight performance improvement. However, the discussed feature representation-based models all show much further improved performance. Especially, PaDiM and PatchCore belong to the state-of-the-art in terms of defect detection and localization. To our knowledge, the AUC scores of 0.991 and 0.981 achieved by the PatchCore method are currently the highest recorded on the MVTec AD data.

Method	L2-AE	L2-AESc	L2-AESc + Stain	SSIM-AE	VAE	GANomaly
Detection	74,0	62,0	89,0	-	-	76,2
Localization	74,0	63,0	81,0	79,0	67,4	-

Table 1: Detection and localization performance measure in AUROC(%) of several discussed reconstruction-based techniques over all classes of the MVTec AD dataset.

Method	MahalanobisAD	SPADE	PaDiM	PatchCore
Detection	95,2	85,5	95,3	99,1
Localization	-	96,5	97,5	98,1

Table 2: Detection and localization performance measure in AUROC(%) of several discussed feature representation-based techniques over all classes of the MVTec AD dataset.

4 Data

This chapter discusses the collection, labeling, processing and dataset splitting of the IRJ images.

4.1 Collection

In order to train a model to detect damages on IRJs, image data needs to be collected. As mentioned previously, image data is widely available since ProRail deploys video inspection of the railway network for maintenance purposes. Every PGO area of the network is inspected twice a year. As with all maintenance tasks, ProRail has outsourced the video inspection to two independent contractors named [REDACTED], who both operate their own video inspection train. The trains are equipped with cameras which generate video data from 10 different angles. In Figure 10 it can be seen that eight cameras are pointed directly towards the track and two at the surrounding environment.

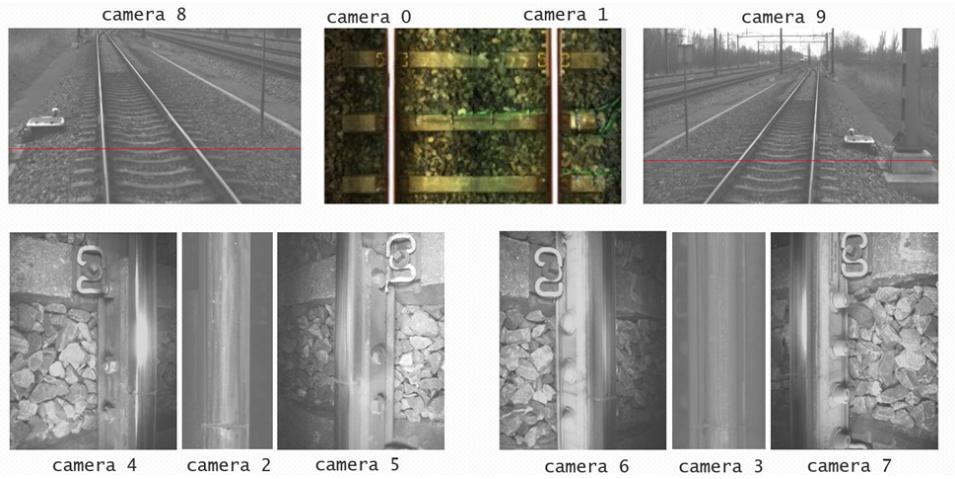
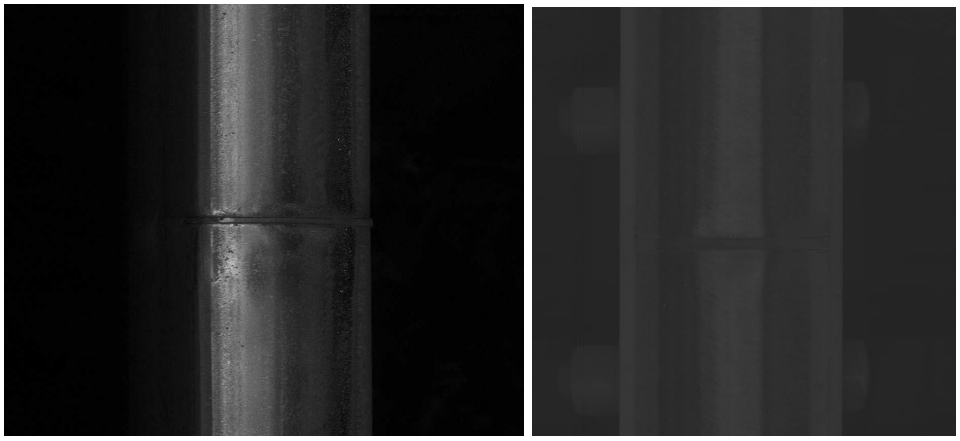


Figure 10: Camera angles of the video inspection train

The data of particular interest for this thesis is generated by the two cameras pointed perpendicular to the running surface of the rail tracks (camera 2 and 3). One is pointed at the left rails and the other is pointed at the right rails. After recording, the gray-scale video data of both cameras is split into non-overlapping frames. The use of these camera angles limits this study to the detection of IRJ and surrounding rail defects as described in Section 2.3.

The imagery provided by the two contractors is not perfectly identical as shown in Figure 11. Firstly, the size of the frames is different as [REDACTED] images have a resolution of 1024x851 pixels, while [REDACTED] images are a smaller 840x850 pixels. Secondly, the contractors presumably use different camera equipment or settings, because [REDACTED] frames have higher contrast and detail than the ones from [REDACTED]. Despite their differences, this study has chosen to utilize both contractors as each PGO area is only captured by one of them. Therefore, it would be more beneficial if our detection method is able to handle both type of images.

The Dutch railway network contains approximately 44.000 IRJs. Fortunately, the AMI Renewal department at ProRail developed a computer vision algorithm to distinguish IRJ



(a) [REDACTED] (1024x851) (b) [REDACTED] (840x850)

Figure 11: Raw image samples from both video inspection train contractors

frames from all other rail track frames. The used data consists of 7154 images of which the distribution between [REDACTED] and [REDACTED] is half-and-half.

4.2 Labelling

The collected images of IRJs are investigated to determine which ones are normal and which ones are damaged. The author was trained by a domain expert on how to recognize each type of IRJ and surrounding rail defect as described in Section 2.3. After training, the images could be classified individually. This was done as an multi-class labelling project in Microsoft Azure ML Studio. Given the collected data and types of defects, the following initial labels were chosen:

- **Functional**: IRJ containing no visible damage.
- **Constriction**: IRJ of which visible end post thickness is less than 3mm.
- **Play**: IRJ of which the end post and adjacent rail head are separated by a gap of at least 1mm.
- **Crumbled**: IRJ of which the end post is fractured, chipped or missing.
- **Spark erosion**: IRJ containing spark erosion on the adjacent rails. Spark erosion regions can be recognized as black areas with white spots directly adjacent to the end post.
- **Squat**: IRJ containing a squat on the surrounding rail surface. Squats are recognized as smooth transitions in reflection but lack clear boundaries.
- **Uncertain**: IRJ of which the condition is uncertain for various reasons. The image could be poorly illuminated, the condition between normal and very slightly defective or the end post is on the extreme edge of the image such that surrounding rail is not entirely visible. Furthermore, it is impossible to accurately label damages such as IRJ indentation from top-down imagery.
- **Other**: Either an uninterrupted rail head, regular joint or rail switch.

The images classified as *Constriction, Spark erosion, Play, Squat* and *Crumbled* will be merged under the *Defect* label. Images classified as *Uncertain* or *Other* will not be included in the datasets. The use of more specific sub-labeling for the defective IRJs gives the opportunity to analyze results per type of defect in a later stage. This could provide insights into how capable our detection model was in retrieving the different defects. It is already evident that spark erosion is most common defect and end post play the least. Examples of labeled images are shown in Figure 12.

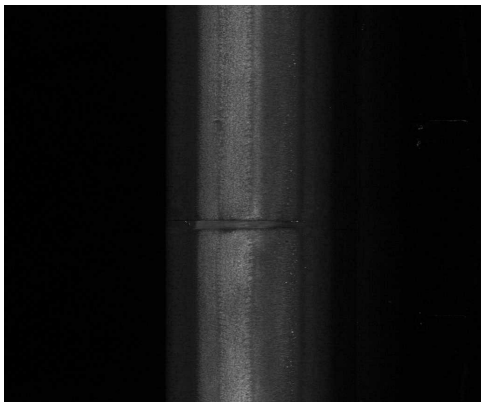
Label	**Number**	**Sub-label**	**Number**
Functional	3079 (72,4%)	Functional	3079
Defect	1173 (27,6%)	Constriction	372
		Spark erosion	466
		Play	93
		Squat	143
		Crumbled	99
-	-	Uncertain	1507
		Other	1395

Table 3: Results of manual data classification

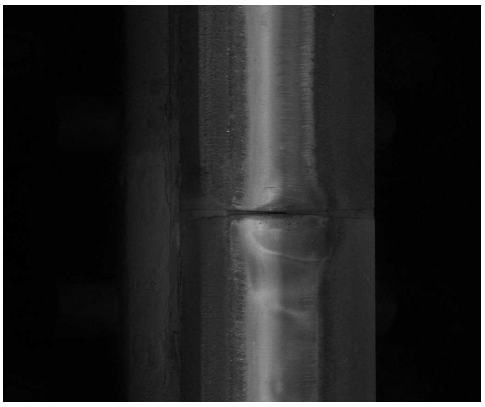
In Table 3 it can be observed that 3079 images are classified as *Functional* and 1173 images as *Defect*. The ratio of normal and defective IRJs is therefore 72,4% against 27,6%, showing a clear class imbalance. This exact imbalance will be used for dataset splitting in Section 4.4.

Data labeling was time-consuming, but even more so rather difficult as evidenced by the relatively high number of *Uncertain* classifications. The is due to factors such as the varying illuminations and reflections of the rails, the differences between [REDACTED] and [REDACTED] images and the subtlety of some defects. Despite extensive documentation on defects, it regularly happens that even domain experts at ProRail disagree about the classification of certain images. In some cases, a clear classification simply does not exist. Some further comments on data labelling:

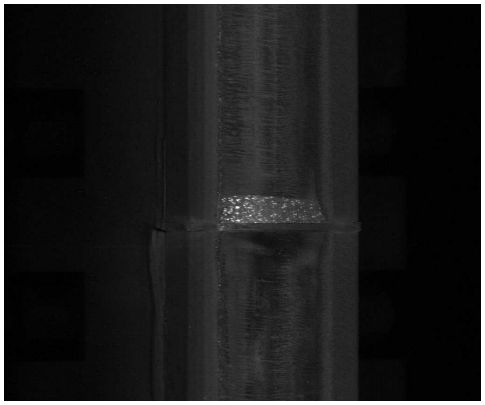
- In case an image is labeled as *Defect*, the sub-label does only highlight the type of defect and not the the severity of the defect.
- When an image shows multiple defects, the sub-label indicates the most severe defect, although this has no influence on the number of defective samples.
- Vertical indentation of IRJs is described in Section 2.3 and on second thought also seen in the data, but not explicitly labeled. For most instances, the camera angle makes it impossible to decide if the depth of the indentation is more than 0.5mm. These images are labeled with the *Uncertain* tag. Severe indentation is often accompanied by constriction of the end post or confused with squats, so indirectly represented.
- Slight corrugation is described and occasionally seen in the data, but not explicitly labeled. In some cases, corrugation is confused with squats.
- Head checks and studs are described but not observed in the data.



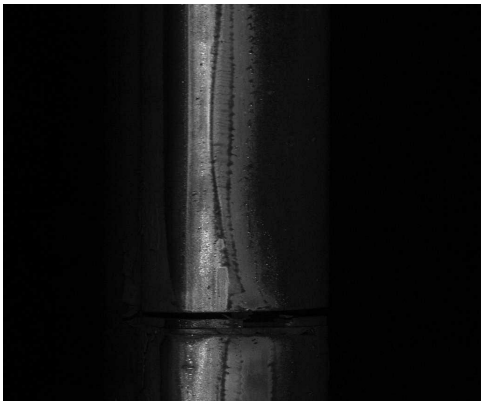
(a) Functional



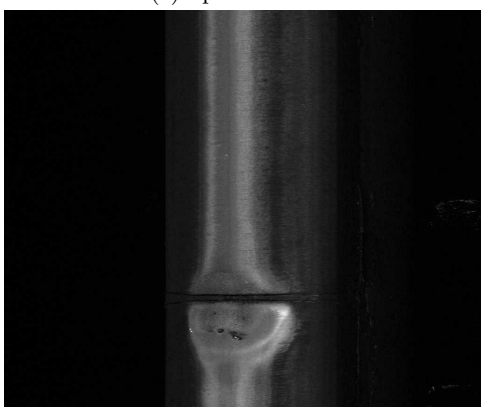
(b) Constriction



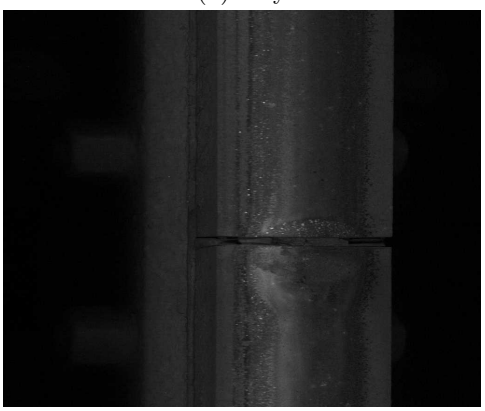
(c) Spark erosion



(d) Play



(e) Squat



(f) Crumbled

Figure 12: Examples of labeled images of functional and defective IRJs

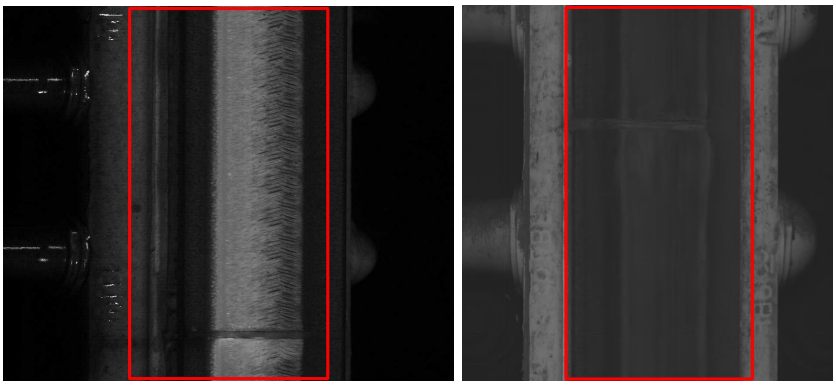
4.3 Processing

The labeled images are cropped to reduce unwanted noise which could affect the detection performance of the semi-supervised learning methods. This study opts for two varying cut-outs to investigate what is most practical and compatible with our approach. The first cut-out extracts the complete rail surface from the image. The second cut-out reduces this area further to merely the IRJ end post region. The images are also slightly enhanced after cropping.

4.3.1 Rail surface extraction

Rail surface extraction is done to prevent the environment of the rail to influence model training and prediction. Although the environment is typically not illuminated, some images clearly show non-relevant IRJ components, such as the joint bars and joint bolts.

To obtain accurate rail cut-out, histogram-based extraction methods of [6] and [5] were implemented but unfortunately were not reliable enough. The non-relevant IRJ components, which should be cut away, trick the methods to crop wider than the actual rail surface. The varying reflections and shadows also caused the methods to often crop narrower than the rail surface. Therefore, rail extraction was performed passively by cropping each image with an identically sized cut-out. The height of this cut-out is 850 pixels similar to the height of the [REDACTED] frames as these are the smallest. The width of the cut-out is estimated with an analysis of the average gray scale value per pixel column and turns out to be roughly 420 pixels. [REDACTED] images are cropped from pixel column 341 to 761 and [REDACTED] imagery from 231 to 651. This simple approach works better than the histogram-based extraction but is unfortunately also not perfect. Even though the rail surface is typically centered in the frame, there is also a slight offset present. As a result, after cropping, some cut-outs may still contain a small amount of non-relevant rail environment. Examples of rail extraction are shown in Figure 13.



(a) [REDACTED] (b) [REDACTED]

Figure 13: A fixed-sized cut-out is used to extract the rail surface and remove the impact of non-relevant IRJ components.

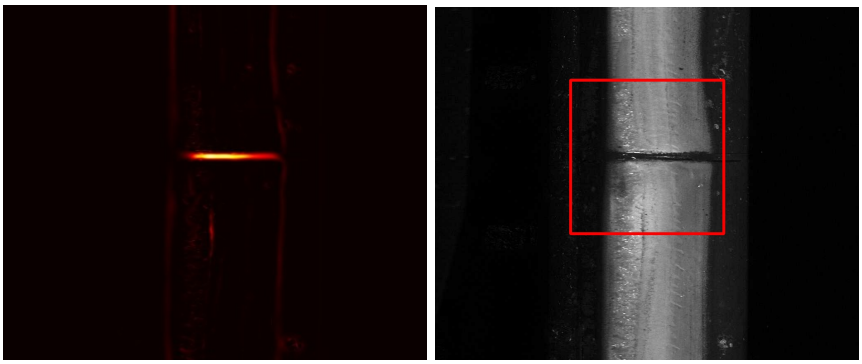
4.3.2 End post extraction

As seen in Figure 12 and 13, the height of the IRJ end post is inconsistent across individual images. This could lead to the semi-supervised method not being able to model the end

post as normality, resulting in many false positive classifications during prediction. To ensure consistency across individual images, the region around the IRJ end post is extracted. By doing so, the end post is always centrally located, making the images more similar to one another than after rail extraction.

In order to accurately crop the IRJ end post region, its vertical position within an image must be determined. One way to accomplish this is by using a ridge detector filter. This type of computer vision method is effective in capturing the interior of elongated image objects, such as IRJ end posts in this case. Our proposed method for end post extraction therefore utilizes the Frangi ridge detector filter, proposed by [40]. This filter effectively highlights the location of the IRJ end post in the image, as shown in Figure 14a. Since the rail is typically centered in the image, it is assumed that the highlighted ridge will be present on the vertical center axis of the image. The highest pixel value on this axis is then assumed to be the location of the IRJ end post. In both [REDACTED] and [REDACTED] images, the region around the found image coordinates is cropped with a cut-out of 350 pixels in height and width, so that the end post is centered in the frame.

This procedure is not possible for 1609 images as the vertical location of the end post varies, and the fixed cut-out is not able to accommodate the end post centrally in frame. This highlights a significant drawback of end post cropping compared to rail cropping, which allows all images to be used. Additionally, 160 images are manually deleted as the Frangi filter sometimes highlighted incorrect ridges, resulting in unsuccessful extraction. Ultimately, the end post extraction process results in 2483 cropped images, as shown in Figure 14b.



(a) Frangi filter (b) End post crop

Figure 14: A fixed-sized cut-out is used to crop the region with the highest activation from the Frangi ridge detector to extract the end post.

4.3.3 Other

The capability of deep learning methods to extract features from raw data makes image enhancement largely unnecessary. Only two operations are performed. First, the rail extracted images are padded left and right to make the image dimension square. Second, all images are resized to a resolution of 256x256 pixels.

4.4 Datasets

The rail extracted images and end post extracted images will be split into independent datasets, respectively the rail dataset and the end post dataset. Both datasets will be

split into training, calibration and test sets. As mentioned, this study will research semi-supervised detection methods, which is why the training sets will contain exclusively IRJ images which are labeled *Functional*. The calibration and test sets contain IRJ images of both labels and are respectively used to obtain a suitable anomaly threshold and to evaluate the final performance of the methodology. These sets have a realistic class proportion of roughly 72% *Functional* and 28% *Defect* to ensures that the methodology can be evaluated to the standards of real-world implementation. Tables 4 and 5 respectively show the distribution of samples over the rail dataset and end post dataset.

Label	**Number**	**Train**	**Calibration**	**Test**
Functional	3079 (72,4%)	1907	586	586
Defect	1173 (27,6%)	-	228	228

Table 4: The rail cropped images are divided into a training set with only functional IRJs and a calibration and test set with both functional and defective IRJs, reflecting the actual class distribution.

Label	**Number**	**Train**	**Calibration**	**Test**
Functional	1766 (72,1%)	1049	358	358
Defect	717 (27,9%)	-	139	139

Table 5: The end post cropped images are divided similarly to the rail cropped images.

As in both cases many functional IRJ images are reserved for the training sets, the realistic distribution of the calibration and test sets has to be created by randomly under-sampling the *Defect* class. This procedure is performed with 42 as the seed. It is verified that after under-sampling, both calibration and test sets contain every type of defect.

5 Methodology

This chapter formulates our semi-supervised defect detection approach. Then it discusses the different reconstruction based methods, feature representation-based methods and scoring functions to implement this approach. Lastly, it describes the evaluation metrics to measure performance.

5.1 Definition: semi-supervised defect detection

Define the following data sets:

- Training set $\boldsymbol{D}_{train} = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ containing N normal samples, where $y_i = 0$ denotes the normal class.
- Calibration set $\boldsymbol{D}_{cal} = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_K, y_K)\}$ of K normal and defective samples where $y_i \epsilon [0, 1]$ denotes a normal or defective sample respectively.
- Test set $\boldsymbol{D}_{test} = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_K, y_K)\}$ of K normal and defective samples where $y_i \epsilon [0, 1]$ denotes the class.

where $N > K$. The samples of each data set are unique and the class distributions within $\boldsymbol{D}_{cal}$ and $\boldsymbol{D}_{test}$ is true to reality.

An image sample $\mathbf{x}$ is a matrix of $\mathbb{R}^{H \times W}$ features (also known as pixels), where $\mathbb{R}\epsilon[0, 255]$ denotes the range of gray-scale values and H and W are the height and width of the matrix. Due to the spatial structure of the image samples, matrix notation is applied for clarity. A bold character is used to denote a sample with a certain spatial structure, while a regular character is used to denote a single value within a sample. Furthermore, when iterating over image samples, the subscript in $\mathbf{x}_i$ denotes the image number. When iterating over the pixels of an image sample, the subscript in x_{ij} denotes the row and column of the pixel. However, when iterating simultaneously over image samples and the pixels within, the image designation becomes a superscript. Thus x_{ij}^k is the pixel at location (i, j) in sample k. The same notation holds for other spatial structures such as image patches $\boldsymbol{p}$ and latent space representations $\boldsymbol{z}$.

The first goal is to learn the underlying normal data distribution of image sample within $\boldsymbol{D}_{train}$ with a model f, such that anomalous samples of $\boldsymbol{D}_{cal}$ and $\boldsymbol{D}_{test}$ can be distinguished from normal samples. Model f should be able to create an anomaly localization map $M(\mathbf{x})$ showing the local abnormality of sample $\mathbf{x}$ in the original image space. An anomaly scoring function $A(\mathbf{x})$ should transform map $M(\mathbf{x})$ into a score denoting the abnormality of sample $\mathbf{x}$ during calibration or testing. Given the fact that model f is adapted to exclusively normal image samples, the anomaly score is assumed to be higher for defective images.

The second goal is to find a cut-off threshold τ which separates normal and defective images on their computed anomaly score. The optimal threshold value is determined by classifying normal and defective samples of $\boldsymbol{D}_{cal}$ at multiple thresholds, such that prediction $\hat{y} = 0$ if $A(\mathbf{x}) < \tau$ and $\hat{y} = 1$ if $A(\mathbf{x}) \geq \tau$. Classification performance is quantified by a given metric c, which compares predictions $\{\hat{y}_1, ..., \hat{y}_K\}$ with the respective ground truths $\{y_1, ..., y_K\}$. The threshold τ maximizing metric c is selected for the combination of model f and score function $A(\mathbf{x})$.

During testing, the combination of model f, anomaly score function $A(\mathbf{x})$ and threshold τ are applied on samples of $\boldsymbol{D}_{test}$ to distinguish normal and defective samples. The results for various combinations of f an $A(\mathbf{x})$ are evaluated and compared to obtain conclusions about their applicability for IRJ defect detection.

5.2 Reconstruction-based methods

This section describes the following reconstruction-based methods for implementation of model f, namely L2-AE, SSIM-AE and Adversarial SSIM-AE. Each method is able to produce a defect localization map $M(\mathbf{x})$ of input image $\mathbf{x}$.

5.2.1 L2-AE

Our first and most basic reconstruction-based localization method is defined as L2-AE. This CAE network is trained to reconstruct normal IRJs by minimizing the Mean Squared Error (MSE) between input and reconstruction with $\mathcal{L}_2$-loss. As discussed in Section 3.3, this method is often applied as a benchmark for defect detection and localization.

The architecture consists of an encoder network E and a decoder network D. In multiple blocks of convolutional layers, batch normalization and non-linear activation, the encoder E encodes input images x to latent space representations $\boldsymbol{z} = G(\mathbf{x})$. The number of features in $\boldsymbol{z}$ is strongly reduced in comparison to the original input $\mathbf{x}$, such that $\hat{\mathbf{x}}$ can not be a one-to-one mapping of $\mathbf{x}$. From this reduced representation $\boldsymbol{z}$ the decoder D creates reconstructions $\hat{\mathbf{x}} = D(\boldsymbol{z})$ similar to the original input $\mathbf{x}$. The decoder mirrors the encoder using transposed convolutional layers instead of standard convolutional layers for trainable upsampling.

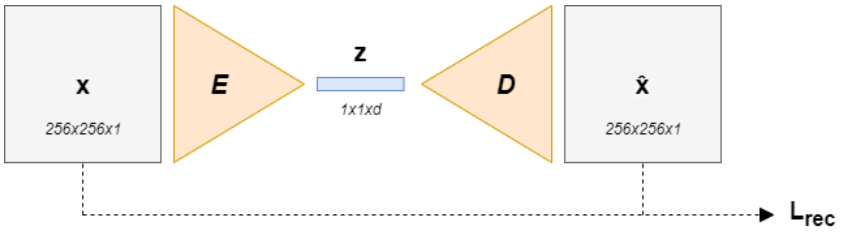


Figure 15: Architecture of the L2-AE and SSIM-AE models

During the training phase, the weights of the convolutional layers in the L2-AE are adapted such that normal samples of $\boldsymbol{D}_{train}$ are reconstructed with the smallest possible error. Low reconstruction errors imply that $\boldsymbol{z}$ will only contain the most descriptive features for normal images. The L2-AE is trained using a $\mathcal{L}_2$-loss function, which measures the reconstruction error between the samples $\mathbf{x}$ and reconstructions $\hat{\mathbf{x}} = D(E(\mathbf{x}))$ as follows:

$$\mathcal{L}_2(\mathbf{x}, \hat{\mathbf{x}}) = \sum_{i=1}^{W} \sum_{j=1}^{H} (\mathrm{x}_{ij} - \hat{\mathrm{x}}_{ij})^2 \tag{1}$$

where W and H are the respective width and height in pixels. As discussed, $\mathcal{L}_2$-loss is a per-pixel loss function which assumes pixel values are independent of each other. During the calibration and testing phases, localization map $M(\mathbf{x})$ is constructed from the pixel-level residuals between input $\mathbf{x}$ and reconstruction $\hat{\mathbf{x}}$:

$$M(\mathrm{x}_{ij}) = (\mathrm{x}_{ij} - \hat{\mathrm{x}}_{ij})^2 \text{ for } 1 \leq i \leq W, 1 \leq j \leq H \tag{2}$$

Large pixel-wise differences in map $M(\mathbf{x})$ indicate the presence of defects. The map is translated into an anomaly score $A(\mathbf{x})$ by using the anomaly scoring functions discussed in Section 5.4.

5.2.2 SSIM-AE

The second reconstruction-based localization method we propose is called SSIM-AE. This model uses the same network architecture as the L2-AE model, but it is trained with a reconstruction loss function based on Structural Similarity (SSIM) instead of MSE. As mentioned in Section 3.3, SSIM was introduced by [29] as a more perceptually based image similarity function, which considers dependencies between local pixel regions of the input and reconstructed image. The SSIM-loss is anticipated to enhance the reconstruction of normal images and negatively impact the reconstruction of defective areas afterwards, thereby improving detection performance.

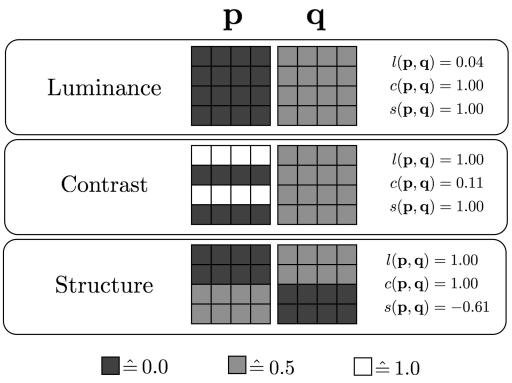


Figure 16: Responsibilities of the luminance, contrast and structure terms deployed by SSIM

According to [29], the SSIM index defines a distance measure which is best applied locally by comparing $K \times K$ patches $\boldsymbol{p}$ and $\boldsymbol{q}$ from two distinct images:

$$SSIM(\boldsymbol{p}, \boldsymbol{q}) = l(\boldsymbol{p}, \boldsymbol{q})^{\alpha} c(\boldsymbol{p}, \boldsymbol{q})^{\beta} s(\boldsymbol{p}, \boldsymbol{q})^{\gamma} \tag{3}$$

where α, β, γ are user-defined weights. The luminance formula $l(\boldsymbol{p}, \boldsymbol{q})$ compares the mean patch intensities μ_p and μ_q, the contrast formula $c(\boldsymbol{p}, \boldsymbol{q})$ compares the variances of the patches σ_p^2 and σ_q^2. Lastly, the structure formula $s(\boldsymbol{p}, \boldsymbol{q})$ considers the covariance σ_{pq} of the two patches. When we consider $\alpha = \beta = \gamma = 1$ for simplicity, the SSIM index between patches $\boldsymbol{p}$ and $\boldsymbol{q}$ is reduced to its common form:

$$SSIM(\boldsymbol{p}, \boldsymbol{q}) = \frac{(2\mu_p\mu_q + c_1)(2\sigma_{pq} + c_2)}{(\mu_p^2\mu_q^2 + c_1)(\sigma_p^2\sigma_q^2 + c_2)} \tag{4}$$

in which terms c_1 and c_2 ensure numerical stability and are typically set to 0.01 and 0.03 respectively. The value of $SSIM(\boldsymbol{p}, \boldsymbol{q})$ lies between -1 and $+1$, where a value of 1 signifies that $\boldsymbol{p}$ and $\boldsymbol{q}$ are exactly the same, a value of -1 represents complete dissimilarity, and a value of 0 indicates no similarity. Figure 16 shows how the luminance, contrast and structure terms in SSIM react to different patches $\boldsymbol{p}$ and $\boldsymbol{q}$. In each scenario MSE would generate a value of 0.25 according to [23].

The SSIM between an image $\mathbf{x}$ and its reconstruction $\hat{\mathbf{x}}$ is calculated by moving a sliding window of size $K \times K$ across both images and computing the SSIM at each location. The SSIM index, as defined in Equation 4, is differentiable, allowing it to be used as a loss function $\mathcal{L}SSIM$ to train the SSIM-AE architecture [23]. To use SSIM as a loss function, it must be subtracted from 1 since $\mathcal{L}SSIM = 0$ is considered a perfect reconstruction. Patches

created by the sliding window on the input image $\mathbf{x}$ and the reconstruction $\hat{\mathbf{x}}$ are denoted as $\boldsymbol{p}$ and $\hat{\boldsymbol{p}}$, respectively.

$$\mathcal{L}_{SSIM}(\mathbf{x}, \hat{\mathbf{x}}) = 1 - \frac{1}{WH} \sum_{i=1}^{W} \sum_{j=1}^{H} SSIM(\boldsymbol{p}_{ij}, \hat{\boldsymbol{p}}_{ij})) \tag{5}$$

where W and H are respectively width and height of input $\mathbf{x}$. During calibration and testing, a localization map $M(\mathbf{x})$ is created by computing the SSIM of corresponding patches between the input and reconstruction image, without aggregating the results:

$$M(\mathrm{x}_{ij}) = 1 - SSIM(\boldsymbol{p}_{ij}, \hat{\boldsymbol{p}}_{ij}) \tag{6}$$

The outputted map is translated into an anomaly score $A(\mathbf{x})$ by using the anomaly scoring functions discussed in Section 5.4.

5.2.3 Adversarial SSIM-AE

Our third reconstruction-based localization method is the Adversarial SSIM-AE. The architecture is essentially similar to SSIM-AE, but during training accompanied by a discriminator network E_{disc}, which task it is to distinguish input images $\mathbf{x}$ from reconstructions $\hat{\mathbf{x}}$. As discussed in Section 3.3, the adversarial training component is anticipated to enhance the reconstruction of normal images. This should negatively impact the networks ability to reconstruct defects and thus improve detection performance.

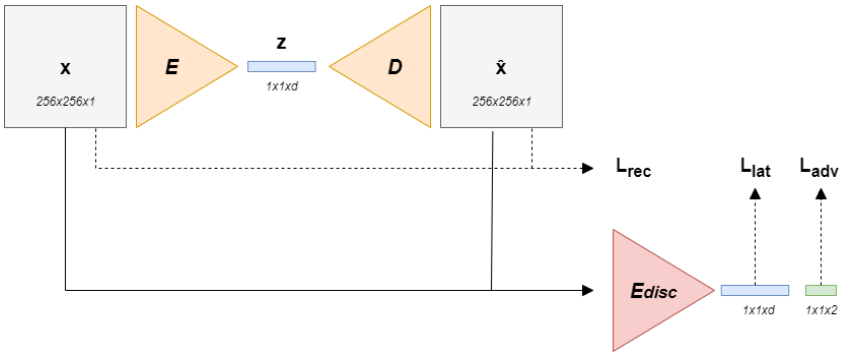


Figure 17: Architecture of the Adversarial SSIM-AE model

The Adversarial SSIM-AE model is trained using a combination of three loss functions aimed at accurately reconstructing normal image samples $\mathbf{x}_{train}$. First, the reconstruction loss $\mathcal{L}_{rec}$ penalizes the distance between the input image $\mathbf{x}$ and output $\hat{\mathbf{x}} = D(E(\mathbf{x}))$, identical to SSIM-AE:

$$\mathcal{L}_{rec}(\mathbf{x}, \hat{\mathbf{x}}) = 1 - \frac{1}{WH} \sum_{i=1}^{W} \sum_{i=1}^{H} SSIM(\boldsymbol{p}_{ij}, \hat{\boldsymbol{p}}_{ij})) \tag{7}$$

where W and H are respectively the width and height of input $\mathbf{x}$. Second, the adversarial loss $\mathcal{L}_{adv}$ is utilized during training to further maximize the normal image reconstruction capability, while discriminator E_{disc} learns to classify original from reconstructed samples. The output of E_{disc} is a probability score that indicates if the provided image is in fact a real input image and not a reconstruction. The objective is to minimize $\mathcal{L}_{adv}$ for SSIM-AE part of the architecture and to maximize it for E_{disc}:

$$\mathcal{L}_{adv}(\mathbf{x}, \hat{\mathbf{x}}) = \log[E_{disc}(\mathbf{x})] + \log[1 - E_{disc}(\hat{\mathbf{x}})] \tag{8}$$

Third, the latent loss $\mathcal{L}_{lat}$ is added with the intention to encode latent representations for input image $\mathbf{x}$ and reconstruction $\hat{\mathbf{x}}$ as similar as possible. According to [31] this ensures that the model becomes capable of producing contextually sound latent representation for normal samples. The contextual loss is measured before the last layer of the discriminator network E_{disc}. Define this part of E_{disc} as E_{short} such that the latent representations are $\boldsymbol{z} = E_{short}(\mathbf{x})$ and $\hat{\boldsymbol{z}} = E_{short}(\hat{\mathbf{x}})$:

$$\mathcal{L}_{con}(\mathbf{x}, \hat{\mathbf{x}}) = \sum_{i=1}^{N} (\boldsymbol{z}_i - \hat{\boldsymbol{z}}_{\boldsymbol{i}}))^2 \tag{9}$$

where N is the number of features in the latent space representations. The overall loss function for training the Adversarial SSIM-AE is created by the weighted sum of the three previously defined loss functions:

$$\mathcal{L} = \omega_{rec}\mathcal{L}_{rec} + \omega_{adv}\mathcal{L}_{adv} + \omega_{lat}\mathcal{L}_{lat} \tag{10}$$

where ω_{rec}, ω_{adv} and ω_{lat} control the impact of each individual loss. During the calibration and testing phases, only the SSIM-AE part of the network is regarded and localization map $M(\mathbf{x})$ is constructed in similar fashion:

$$M(\mathrm{x}_{ij}) = 1 - SSIM(\boldsymbol{p}_{ij}, \hat{\boldsymbol{p}}_{ij}) \tag{11}$$

The map is translated into an anomaly score $A(\mathbf{x})$ by using the anomaly scoring functions discussed in Section 5.4.

5.3 Feature representation-based methods

This section outlines the implementation of two feature representation-based methods for model f, PaDiM-R18 and PaDiM-WR50. These methods differ in their use of pre-trained network backbones, with PaDiM-R18 using ResNet-18 and PaDiM-WR50 using Wide ResNet-50. Both methods are capable of generating a defect localization map $M(\mathbf{x})$ from an input image $\mathbf{x}$, which is a key feature of the PaDiM framework. As described in Section 3.3, the PaDiM framework is one of the best performing methods on MVTec AD and more efficient than other models as it does not rely on the K-Nearest Neighbor (KNN) algorithm.

5.3.1 PaDiM

The Patch Distribution Modelling (PaDiM) framework was introduced by [1]. It uses pre-trained Convolutional Neural Networks (CNNs) to extract features from normal image patches and model their distribution using multivariate Gaussians (MVGs). As discussed in Section 3.3, the pre-trained CNNs are trained on ImageNet, which includes images that are intrinsically different from the IRJ images. However, the CNNs can still generate meaningful feature representations that be used to model normality and afterwards distinguish defective IRJs. The overall structure of the PaDiM framework is depicted in Figure 18.

Feature extraction and concatenation PaDiM extracts feature representations from the first three layers of the CNN to model multivariate Gaussian distributions. Each patch of the input image $\mathbf{x}$ is associated with a corresponding vector in these representations as shown in Figure 18. The activation vectors from the different layers are concatenated to contain both local and global context, resulting in a lower-dimensional representation for nearby pixels in the input image. As a result, the input image $\mathbf{x}$ can be divided into a

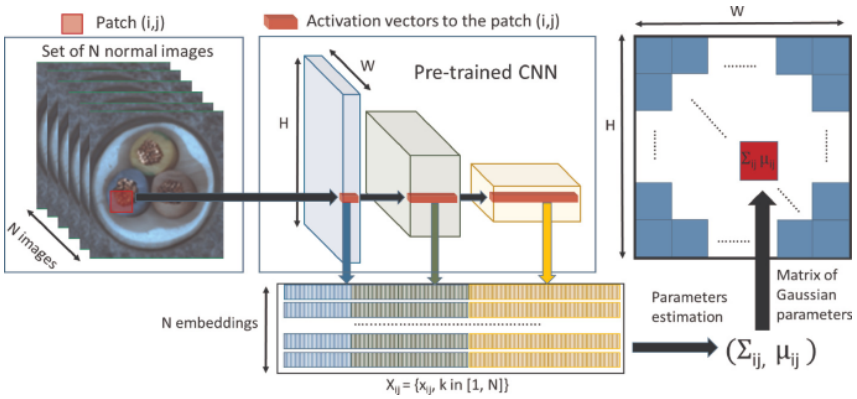


Figure 18: The PaDiM framework models the embedding vectors of image patches as parameters of multivariate Gaussian distributions during training.

grid of patches, each associated with a representation vector $\boldsymbol{z}_{ij}$, where i and j indicate the location of the patch in the input image.

Dimensionality reduction The authors of [1] propose to randomly reduce the number of dimensions in the patch representation vectors to d to eliminate the redundancy of information. This random dimensionality reduction method is more efficient than Principal Component Analysis (PCA), while the performance of the model is still maintained.

Distribution modeling During training, normality is modeled by computing sets of representation vectors for similar patch positions in each sample of $\boldsymbol{D}_{train}$. The set at patch location (i, j) is $\boldsymbol{Z}_{ij} = \{\boldsymbol{z}_{ij}^k, k\epsilon[1, N]\}$ where N is the number of training samples. Each set is assumed to be generate by a MVG distribution $\mathcal{N}(\mu_{ij}, \Sigma_{ij})$ where μ_{ij} is the sample mean of $\boldsymbol{Z}_{ij}$ and where Σ_{ij} is the sample covariance:

$$\Sigma_{ij} = \frac{1}{N-1}\sum_{k=1}^{N}(\boldsymbol{z}_{ij}^k - \mu_{ij}^k)(\boldsymbol{z}_{ij}^k - \mu_{ij}^k)^T + \epsilon I \tag{12}$$

in which the term ϵI ensures that the sammple covariance is full rank and invertible. After μ_{ij} and Σ_{ij} are computed for each set $\boldsymbol{Z}_{ij}$, we end up with a matrix of Gaussian parameters as shown in Figure 18.

Mahalanobis distance During calibration and testing, the abnormality of each image patch is scored with Mahalanobis distance to create localization map $M(\mathbf{x})$. The scoring metric expresses the distance between patch embedding $\boldsymbol{z}_{ij}$ and the modeled $\mathcal{N}(\mu_{ij}, \Sigma_{ij})$ at that position:

$$M(\boldsymbol{z}_{ij}) = \sqrt{(\boldsymbol{z}_{ij} - \mu_{ij})^T \Sigma_{ij}^{-1} (\boldsymbol{z}_{ij} - \mu_{ij})} \tag{13}$$

The matrix $M(\boldsymbol{z})$ is then resized to the resolution of the input image $\mathbf{x}$ using bilinear interpolation, resulting in the localization map $M(\mathbf{x})$. Finally, the anomaly map is transformed into an anomaly score $A(\mathbf{x})$ through the use of the anomaly scoring functions outlined in Section 5.4.

5.4 Anomaly scoring

This section describes the anomaly scoring functions applied to transform localization map $M\mathbf{x}$) into a anomaly score $A(\mathbf{x})$, indicating the overall abnormality of image $\mathbf{x}$. The combination of each anomaly scoring function and localization method creates a multiple of detection methods. We propose the following scoring functions: mean scoring (A_1), standard deviation scoring (A_2) and maximum patch scoring (A_3).

5.4.1 Mean

Mean scoring (A_1) returns the average value of the pixels in localization map $M\mathbf{x}$). This function is the most straightforward way to score an image and assumes that defective IRJs have large defective areas which strongly show up in an anomaly map. As a result, it may struggle to identify smaller and less pronounced defects.

$$A_1(\mathbf{x}) = \frac{1}{W \times H} \sum_{i=1}^{W} \sum_{j=1}^{H} M(\mathrm{x}_{ij}) \tag{14}$$

5.4.2 Standard deviation

The standard deviation scoring (A_2) calculates the spread of pixel values in the anomaly map $M(\mathbf{x})$. This scoring method is considered to be slightly better at handling inaccurate anomaly maps than mean scoring. For instance, if the score map of a functional IRJ has equally activated pixels, mean scoring would assign a high anomaly score due to the substantial brightness of each pixel in $M(\mathbf{x})$. However, standard deviation scoring would assign a low anomaly score as there is limited variation between pixel values.

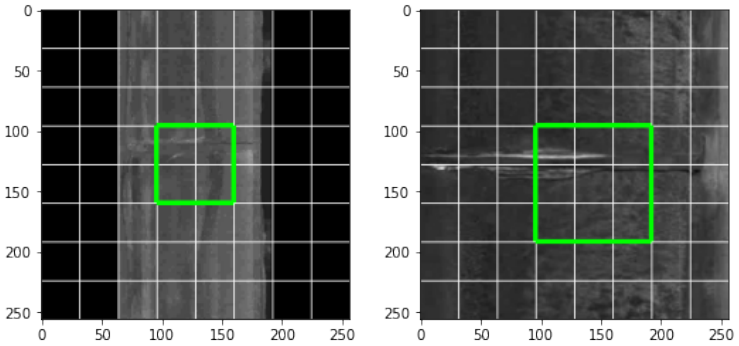
$$A_2(\mathbf{x}) = \sqrt{\frac{1}{W \times H - 1} \sum_{i=1}^{W} \sum_{j=1}^{H} (M(\mathrm{x}_{ij}) - A_1(\mathbf{x}))^2} \tag{15}$$

5.4.3 Maximum patch

Maximum patch scoring (A_3) calculates the average of the highest scoring region of pixels in the localization map $M(\mathbf{x})$. This method involves setting a patch size K for the width and height of the patch and a step size S for the sliding operation. Unlike mean and standard deviation scoring, it assumes that the defect is concentrated in a specific region of the localization map, which is supported by the fact that all labelled defects are clusters of pixels. In comparison, mean and standard deviation scoring may generate high anomaly scores if defects in an IRJ and surrounding rails are unrealistically spread out over a large area. The formula for max patch scoring is defined as follows, with $\boldsymbol{p}$ being an KxK image patch of $\mathbf{x}$:

$$A_3(\mathbf{x}) = max(M(\boldsymbol{p}_i)_{1 \leq i \leq (\frac{W}{S} - K + S)^2}) \tag{16}$$

in which i is the patch number and W is the width of $\mathbf{x}$ assuming that it has a square dimension. The patch size K is set to 64 pixels and step size S is 32 pixels for rail cropped images and K is increased to 96 pixels for end post cropped images. The maximum patch score is then determined by the maximum value of 49 patch averages for rail cropped images and 36 patch averages for end post cropped images. The reason for using bigger patches for end post crops is that the closer view of the IRJ allows defects to appear larger. The size of these patches in comparison to the corresponding input images is visualized in Figure 19.



(a) Rail cropped image (b) End post cropped image

Figure 19: The patch size of the A_3 scoring function varies based on the dataset. The visualization uses input images $\mathbf{x}$ to show the patch size, but in actual scoring, the localization maps $M(\mathbf{x})$ are used.

5.4.4 Normalization

The set of anomaly scores generated by each invidudal scoring function is defined as $A = \{A(\mathbf{x}_i), 1 \leq i \leq N\}$ and as a final step during calibration and testing, normalized to scale within the range of [0,1]. This creates the set of normalized anomaly scores A', which is used to rank images from normal to more anomalous:

$$A'(\mathbf{x}) = \frac{A(\mathbf{x}) - min(A)}{max(A) - min(A)} \tag{17}$$

5.5 Evaluation metrics

The aim of the binary classification task, as stated in Section 5.1, is to correctly predict the class of unseen IRJ image samples $\mathbf{x}$. The results of this task can be summarized using a confusion matrix with the following categories:

- TP: the number of defective IRJs correctly classified as *Defect*
- FP: the number of functional IRJs wrongly classified as *Defect*
- TN: the number of functional IRJs correctly classified as *Functional*
- FN: the number of defective IRJs wrongly classified as *Functional*

Different evaluation metrics can be derived from these classification outcomes to assess model performance in the calibration and test phases. However, since our data classes are imbalanced, some metrics like accuracy are not suitable, as they would give misleading results. This is because accuracy measures the ratio of correct predictions over all predictions, and simply classifying all samples as the majority class would result in high accuracy, even though the minority class predictions are wrong. Hence, this study opts for the use of AUC and F1-score, which are more appropriate for imbalanced data.

5.5.1 ROC curve and Area Under Curve

The Receiver Operating Characteristic (ROC) curve is a graph showing the performance of the binary classifier for different thresholds. The curve is generated by plotting two metrics on either axis, namely the true positive rate (TPR) and the false positive rate (FPR):

$$TPR = \frac{TP}{TP + FN} \tag{18}$$

$$FPR = \frac{FP}{FP + TN} \tag{19}$$

The curve shows at what rate lowering of the threshold will lead to more positive classifications, increasing the number of true and false positives and vice versa.

The Area Under the Curve (AUC) aggregates the information in the ROC curve and provides information about the classifier's performance for all possible thresholds. The outcome, which ranges between 0 and 1, can be seen as the degree to which the samples of the two classes can be separated. AUC is less sensitive to data imbalance because it is invariant to the threshold value. As our approach requires a threshold, the AUC score will only serve as an indication of model performance during the calibration phase.

5.5.2 Precision, Recall and F1

Precision, recall and F1-score are metrics to evaluate binary classification performance for a specified threshold. First, precision is the ratio of true positive classifications over the total number of positives classifications and therefore describes the classifier's ability to not unnecessarily label functional IRJs as defective:

$$Precision = \frac{TP}{TP + FP} \tag{20}$$

Second, recall is the ratio of true positive predictions over the total number of positive instances and thus shows the classifier's ability to retrieve all defective IRJs:

$$Recall = TPR = \frac{TP}{TP + FN} \tag{21}$$

Lastly, F1-score tries to balance the trade-off between precision and recall by being the harmonic mean of the two metrics. his makes F1-score less sensitive to imbalanced data than accuracy.

$$F_1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)} = 2 \times \frac{precision \times recall}{precision + recall} \tag{22}$$

All three metrics are scored on a scale from 0 to 1, with 1 being the perfect score. For the evaluation of our detection methods, F1-score is considered a suitable choice due to its capability of incorporating information from both precision and recall and its resistance to data imbalance. The metric will therefore serve to establish optimal thresholds for each detection method on the calibration data and rank the performance of each method and calculated threshold on the test data. However, precision and recall are still significant metrics to analyze, as the aim of our solution is to minimize manual inspection and accurately detect all defective IRJs. Precision roughly indicates the reduction in manual inspection, while recall represents the effectiveness of the model in identifying defects.

6 Experiments

This chapter discusses the experimental setup, which regards the parameters of the models the procedures of training, calibration and testing. Furthermore, it shares the results of each procedure.

6.1 Experimental setup

This study will evaluate the performance of all possible combinations of dataset, localization method, and scoring function. Each combination will be considered as a distinct detection approach, with a total of 30 possibilities due to the use of two datasets, five models, and three scoring functions, as shown in Table 6.

Dataset	Method	Scoring
Rail crop	L2-AE	Mean (A_1)
End post crop	SSIM-AE	Standard deviation (A_2)
	Adversarial SSIM-AE	Maximum patch (A_3)
	PaDiM-R18	
	PaDiM-WR50	

Table 6: Each approach is a different combination of dataset, method and scoring function.

6.1.1 Parameters of reconstruction-based methods

The architectures of the reconstruction-based methods, described in Section 5.2, are based on the GANomaly model by [27], which itself is based on the DCGAN architecture by [41]. Table 7 shows the outline of encoder network E of the L2-AE, SSIM-AE and Adversarial SSIM-AE models. The network is composed of seven blocks of convolutional layers, with batch normalization applied after layers 2 to 6. Layers 1 to 6 are furthermore followed by the Leaky Rectified Linear Unit (LeakyReLU) function for non-linear activation:

$$LeakyRelu(z, a) = \begin{cases} z \text{ if } z > 0 \\ az \text{ if otherwise} \end{cases} \tag{23}$$

in which z is an arbitrary value from a feature map output and a is the slope set to 0.2. Trough all layers, the encoder networks reduces an input sample $\mathbf{x}$ of 256x256x1 pixels down to $d = 2048$ features in latent space $\boldsymbol{z}$. This implies that the input is compressed to approximately 3% of its original size.

The architecture of encoder E is copied over to discriminator network E_{disc} in the Adversarial SSIM-AE model. The only difference is that the last layer in E_{disc} is followed by a sigmoid activation function. This transforms the embedding of the input sample into a probability score, indicating the likelihood of the sample being real:

$$sigmoid(z) = \frac{1}{1 + e^{-z}} \tag{24}$$

The network design of E is inverted to create the decoder network D in the L2-AE, SSIM-AE and Adversarial SSIM-AE models. To perform trainable upsampling, the convolutional layers are replaced with transposed convolutional layers. These layers generate an output feature map with a larger spatial dimension than the input feature map. Given the kernel size k, padding size p and stride s of a standard convolutional layer, the transposed convolutional layer inserts $r = s - 1$ zeros between each row and column of the input map. It then pads

Layer	Output Size	Parameter		
		Kernel (k)	Stride (s)	Padding (p)
Input	256x256x1			
Conv1	131x131x4	4	2	4
Conv2	65x65x8	4	2	1
Conv3	32x32x16	4	2	1
Conv4	16x16x32	4	2	1
Conv5	8x8x64	4	2	1
Conv6	4x4x128	4	2	1
Conv7	1x1xd	4	1	0

Table 7: Outline of encoder network E, which is also applied as discriminator network E_{disc}. The inverse version of E becomes decoder network D.

the input with $p^{'} = k - p - 1$ zeros and carries out a standard convolution with a modified stride $s^{'} = 1$. After upsampling the latent space $\boldsymbol{z}$ to the original resolution of the input $\mathbf{x}$, the final layer of the decoder applies a tanh activation function:

$$tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{25}$$

Lastly, regarding the loss functions of the SSIM-AE and Adversarial SSIM-AE models, the window size for SSIM-loss is set to $K = 11$, in line with [29]. The weights in the combined adversarial loss function are $\omega_{rec} = 10$, $\omega_{adv} = 1$, and $\omega_{lat} = 1$.

6.1.2 Parameters of feature representation-based methods

The implementation of the PaDiM framework will follow the original form as described by [1]. As stated in Section 5.3, this framework will be used for the PaDiM-R18 and PaDiM-WR50 models, which vary in the underlying pre-trained CNNs and the number of randomly selected dimensions from the extracted patch embeddings, as shown in Table 8. The random dimensionality reduction is done with a seed of 42. The other specified parameters are $\epsilon = 0.01$ in the sample covariance formula and standard deviation $\sigma = 4$ of Gaussian blurring, which is applied after resizing the localization map $M(\mathbf{x})$ with bilinear interpolation.

Model	Backbone	Patch representation	
		Dimensions	Selected (d)
PaDiM-R18	ResNet-18	448	100
PaDiM-WR50	Wide ResNet-50	1792	550

Table 8: The PaDiM-R18 and PaDiM-WR50 models use different underlying feature extractor and number of randomly selected embedding vectors.

6.1.3 Training

Each reconstruction-based method and feature representation-based method is trained on exclusively functional IRJ images from the rail and end post crop training sets. This training procedure thus consists of 10 training runs.

The L2-AE, SSIM-AE and Adversarial SSIM-AE models learn to reconstruct input images from scratch. The networks are optimized using Adam with an initial learning rate $lr = 2e^{-4}$

and momentum terms $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The training set of both rail and end post crops is randomly partitioned into 90% actual training data and 10% validation data with a batch size of 32 samples. During training, the network weights are only saved when the loss over the validation samples is lowered. Training stops if the maximum number of epochs is reached or if the validation loss is not lowered for a specified number of epochs. The early stopping procedure can saves time and counteract overfitting on the training data. On the rail crop data, the networks are trained for 50 epochs with early stopping after 15 epochs, on the end post crop data, these values are respectively 100 and 20 epochs. The higher number of epochs is justified by the smaller training set size of the end post crop data. The PaDiM-R18 and PaDiM-WR50 models are based on pre-trained ResNets and do therefor not use any hyperparameters. These models will be trained on the original number of samples in the rail and end post training sets.

All models are implemented with the PyTorch and NumPy libraries for Python. The training procedures will be carried out on a GPU compute unit in Microsoft Azure ML studio which uses NVIDIA's Tesla K80 GPU and 56 gigabytes of RAM memory. Each model is after training able to output a localization map showing the abnormal areas of an unseen IRJ image.

6.1.4 Calibration

During calibration, the trained methods are paired with each scoring function to generate anomaly scores for the IRJ images in the two calibration sets. The ROC-curve, AUC score, and score distribution of each approach are analyzed to indicate its class separation ability during testing. Afterwards, the optimal threshold τ for each of the 30 approaches is determined by maximizing the F1-score, considering the calculated anomaly scores and actual ground truths. Figure 20 illustrates the score distribution of two classes and the impact of the threshold on the number of binary classification results. If the classes cannot be separated completely based on the anomaly scores, it is be possible to set a threshold without encountering classification errors.

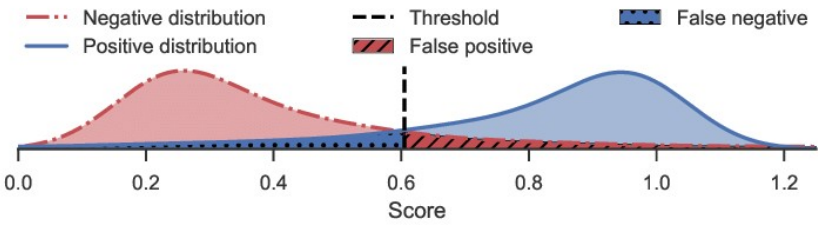


Figure 20: Effect of the threshold on the binary classification results

6.1.5 Testing

In the testing phase, each of the 30 approaches will be applied using the determined thresholds to score and classify unseen IRJ images of the test sets. The results will be presented in terms of the F1-score, recall, and precision outcomes of each approach. The approach with the highest F1-score will undergo further evaluation to determine its ability to identify individual defect types as defined by the sub-labels discussed in Section 4.2.

6.2 Results

6.2.1 Training

This section will first examine the observations made during the training of the reconstruction-based methods. The loss curves for each method are shown in Figure 30 in Appendix A.1. For the training set of rail crop images, all three reconstruction models were trained for 50 epochs and displayed continuous improvement in validation loss. The L2-AE achieved the lowest validation loss at the final epoch, while the SSIM-AE reached its lowest validation loss at epoch 41 and the Adversarial SSIM-AE at epoch 45. For the training set of end post crop images, the reconstruction models were trained for 100 epochs with early stopping after 20 epochs without improvement. The L2-AE and Adversarial SSIM-AE were trained for the full number of epochs and achieved the best validation loss at epoch 83 and 97 respectively. However, the SSIM-AE converged faster and stopped training after 31 epochs as it did not show any improvement in validation loss after epoch 11. The SSIM-AE training loss curve in Figure 30d shows a relatively linear trajectory, which suggests that the model started with good weight initialization, as its reconstruction ability is comparable to the other methods.

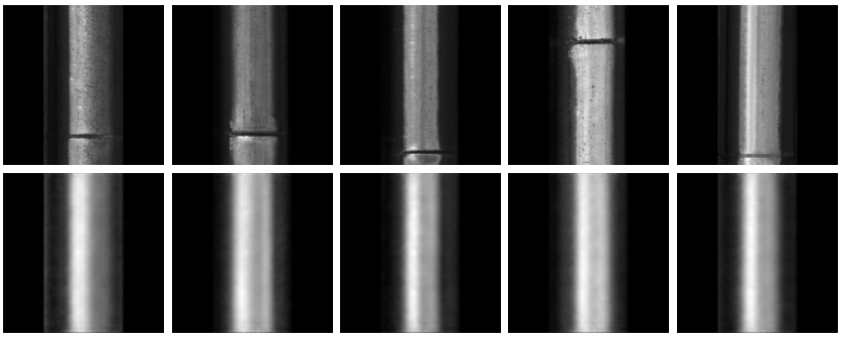


Figure 21: The Adversarial SSIM-AE is able to replicate (bottom) the metallic rail surface of the input rail crop images (top).

The reconstruction-based methods are all able to broadly reconstruct the input images after training. Figures 21 and 22 show sample reconstructions of rail and end post crops from the Adversarial SSIM-AE respectively. The rail crop reconstructions mostly capture metallic reflections and shadows, but small surface imperfections are lost. The end post crop reconstructions are more detailed as they provide a closer perspective of the IRJs, but they are still blurry and do not accurately reconstruct small surface irregularities. The most apparent difference between the image crops is that the IRJ end post is only reconstructed in end post crops, as expected due to its central positioning within the frames. Additionally, it is noted that models trained with the SSIM-loss function occasionally produce white artifacts in some of the reconstructions, as seen in the rightmost example of Figure 22.

The training of the feature representation-based models is less opaque than the reconstruction-based methods for learning normality. As a result, there is less to observe and compare. However it can be commented that the training of both PaDiM models is much quicker compared to the AE models. While the AE methods typically take 1 to 2 hours to complete on either training set, with the exception of the SSIM-AE on the end post crops, the PaDiM-R18 and PaDiM-WR50 finish within just 10 and 20 minutes on rail and end post crop training data. The only problem was that the hardware crashed multiple times during the training of the PaDiM-WR50 model on the rail crop images. To resolve this issue, the complexity of the training process had to be reduced either by decreasing the number of

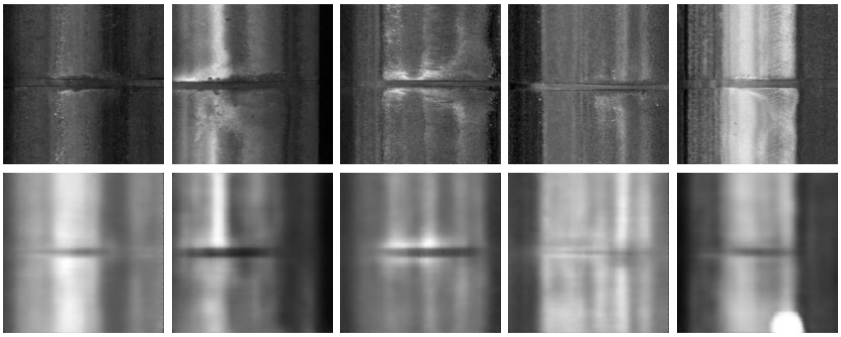


Figure 22: The Adversarial SSIM-AE reconstructs (bottom) both the metallic rail structure and IRJ end posts of the input end post crop images (top). The use of SSIM-loss can lead to reconstruction artifacts, as seen in the rightmost image.

randomly selected embedding vectors or the number of training samples. The latter was preferred as we aimed to avoid changing model parameters. The number of training samples was reduced from 1907 to 1000.

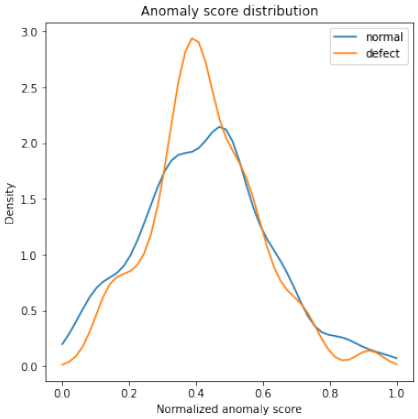
In short:

- The SSIM-AE on the end post crop dataset is the only reconstruction-based method which stopped early during training.
- The reconstruction-based methods are only able to reconstruct the end post in the end post cropped images.
- The reconstruction methods applying SSIM-loss sometimes produce reconstruction artifacts.
- The PaDiM methods are more efficient in learning normality than the AE methods as they are based on pre-trained networks.
- The PaDiM-WR50 method only finishes training on the rail crop dataset after reducing the number of training samples from 1907 to 1000.

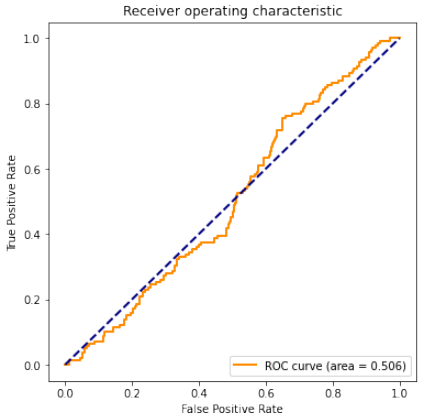
6.2.2 Calibration

In the calibration phase, the ability of each approach to separate functional and defective IRJs is determined by examining the AUC scores and score distributions. The results, presented in Tables 14 and 15 in Appendix A.2, show that the AUC scores range from 0.5 to 0.75, indicating that class separation ability ranges from non-existent to substantial. The results are illustrated by two extreme scoring approaches. The Adversarial SSIM-AE with A_1 scoring on the end post crops (Figure 23a and 23b) results in an AUC score of 0.506 and does not separate functional and defective IRJs, making predictions no better than random guessing for any threshold. Conversely, the PaDiM-WR50 with A_2 scoring on the end post crops (Figure 23c and 23d) produces an AUC of 0.746, creating substantial separation between functional and defective IRJs. However, the still substantial overlap between the score distributions implies that false positives and false negatives will occur for any threshold.

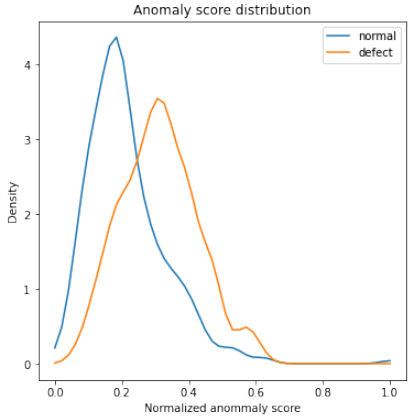
Following this inspection, the thresholds for each approach are determined by optimizing the F1-score over the calibration images. The found thresholds and accompanying F1-scores are presented in Table 9 and 10 for the calibration set of rail crops and end post crops. The



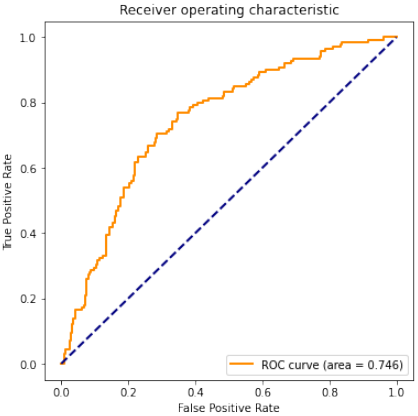
(a) Score distribution of Adversarial SSIM-AE with A_1 scoring function



(b) ROC-curve of Adversarial SSIM-AE with A_1 scoring function



(c) Score distribution of PaDiM-WR50 with A_2 scoring function



(d) ROC-curve of PaDiM-WR50 with A_2 scoring function

Figure 23: The performance of two approaches is compared through their anomaly score distributions and ROC-curves. The Adversarial SSIM-AE with A_1 scoring (a-b) does not effectively differentiate between normal and defective IRJs, while the PaDiM-WR50 with A_2 scoring (c-d) shows substantial differentiation.

use of F1-score for thresholding, results in relatively low thresholds for approaches creating completely overlapping score distributions, such as seen in Figure 23a. Approaches creating more separated class distributions receive a higher threshold, typically at the intersection of score distributions, such as seen in Figure 23c. Obviously it holds that more class separation results in higher F1-score.

	L2-AE	SSIM-AE	Adv. SSIM-AE	PaDiM-R18	PaDiM-WR50
A_1	0.090 (0.454)	0.084 (0.446)	0.196 (0.449)	0.261 (0.458)	0.176 (0.450)
A_2	0.102 (0.468)	0.324 (0.456)	0.340 (0.450)	0.348 (0.461)	0.286 (0.491)
A_3	0.069 (0.469)	0.254 (0.457)	0.318 (0.468)	0.420 (0.455)	0.370 (0.542)

Table 9: The optimal thresholds for each approach on the rail crop calibration set with accompanying F1-scores stated between brackets.

	L2-AE	SSIM-AE	Adv. SSIM-AE	PaDiM-R18	PaDiM-WR50
A_1	0.065 (0.459)	0.140 (0.448)	0.115 (0.450)	0.280 (0.449)	0.412 (0.509)
A_2	0.048 (0.461)	0.341 (0.473)	0.395 (0.482)	0.302 (0.550)	0.232 (0.578)
A_3	0.066 (0.455)	0.176 (0.449)	0.344 (0.472)	0.165 (0.485)	0.169 (0.499)

Table 10: The optimal thresholds for each approach on the end post crop calibration set with accompanying F1-scores stated between brackets.

In short:

- The different approaches show none to substantial ability to separate functional and defective IRJs on their calculated anomaly scores. The AUC scores range from 0.506 to 0.746.
- The thresholds are obtained by optimizing for F1-score. Completely overlapping score distributions lead to low thresholds and low F1-scores, more separated distributions are thresholded at the intersection point and result in higher F1-scores.

6.2.3 Testing

Each approach is combined with its optimal threshold to classify the functional and defective IRJs in the test set. The F1-scores on the rail and end post crop test sets are respectively presented in Tables 11 and 12. The related precision and recall scores are presented in Tables 16 and 17 in Appendix A.3.

The combination of the PaDiM-WR50 method with A_3 scoring yields the highest F1-score of 0.498 on rail crop images. It is worth noting that this specific method was trained with a reduced number of training samples as discussed in Section 6.2.1. The PaDiM-R18 with the same scoring function actually returns a higher score of 0.502, but should be disregarded as the localization map analysis in Section 7.1 reveals that this approach incorrectly localizes. The aggregated results of the individual methods and scoring functions provide more insights.The Adversarial SSIM-AE is the highest scoring reconstruction-based method with an aggregated F1-score of 0.451. The highest scoring feature representation-based method is the PaDiM-WR50 with an overall F1-score of 0.471. The rail crops appear to be best evaluated with the A_3 scoring function, yielding an aggregate F1-score of 0.483, significantly higher than A_1 and A_2 scoring.

	L2-AE	SSIM-AE	Adv. SSIM-AE	PaDiM-R18	PaDiM-WR50	Average
A_1	0.421	0.442	0.441	0.433	0.434	0.434
A_2	0.423	0.430	0.428	0.467	0.480	0.446
A_3	0.470	0.460	0.484	**0.502**	**0.498**	**0483**
Average	0.438	0.444	**0.451**	0.467	**0.471**	**0.454**

Table 11: F1-scores for each model scoring combination on the rail crop test set. The bold faced results are highlighted in the text.

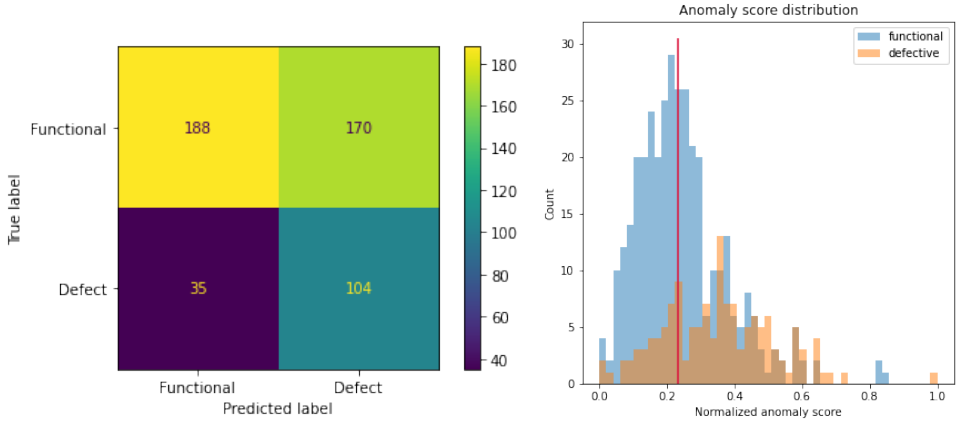
The highest ranking model scoring combination on end post crop images is the PaDiM-WR50 with A_2 scoring, which has a F1-score of 0.507. The aggregated results of the different methods and scoring functions show similarities to those from the rail crop test set. The Adversarial SSIM-AE and PaDiM-WR50 remain the best ranking reconstruction-based and embedding-based methods with F1-scores of 0.461 and 0.488, respectively. These scores are both higher than their scores on the rail crop images. The most appropriate scoring function

for end post crops appears to be A_2 with an aggregated F1 of 0.485.

	L2-AE	SSIM-AE	Adv. SSIM-AE	PaDiM-R18	PaDiM-WR50	Average
A_1	0.428	0.461	0.444	0.460	0.472	0.453
A_2	0.473	0.460	0.490	0.494	**0.507**	**0.485**
A_3	0.387	0.452	0.449	0.484	0.484	0.451
Average	0.429	0.458	**0.461**	0.479	**0.488**	**0.463**

Table 12: F1-scores for each model scoring combination on the end post crop test set. The bold faced results are highlighted in the text.

The average F1-score for the complete test set is 0.454 for rail crops and 0.463 for end post crops, indicating better model performance on the more zoomed in end post crops. Additionally, it observed in Tables 11 and 12 that the L2-AE has the lowest aggregated F1-score, while the other models rank progressively higher. The SSIM-loss and adversarial training improve performance compared to the standard L2-AE. The PaDiM-R18 performs better than the Adversarial SSIM-AE, and the PaDiM-WR50 is the best performing method on both crop types.



(a) Confusion matrix

(b) Score histogram

Figure 24: The detection results of PaDiM-WR50 with A_2 scoring on end post crop test set, which is the highest scoring approach.

Evaluation of highest scoring approach The approach with the highest F1-score is shown to be the PaDiM-WR50 with A_2 scoring on end post crops. Table 17 in Appendix A.3 shows that this approach scores a recall of 0.755 and precision of 0.382. The confusion matrix in Figure 24a depicts the exact true and false detection numbers. It reveals that this approach is able to retrieve over 75% of the defective IRJs, but also falsely classifies 45% of functional IRJs as defective. Furthermore, the truly defective IRJs are categorized by their respective sub-label, as described in Section 4.2. Table 13 shows the number of correctly and incorrectly classified defective IRJs per type of defect. It turns out that this specific PaDiM-WR50 approach is able to retrieve 88.2% of the IRJs with spark erosion regions and 95.6% of IRJs with squats. Most false negatives are made on IRJs of which the end post is either constricted, crumbled or loose. These categories of defective IRJs are retrieved 55.8%, 54.5% and 63.6% of the time, respectively.

	Constriction	Spark erosion	Squat	Crumbled	Play	Total
Correct	24	45	22	6	7	104
Incorrect	19	6	1	5	4	35
Recall	0.558	0.882	0.956	0.545	0.636	0.755

Table 13: The number of correct and incorrect detections of defective IRJs by the best-performing approach, categorized by type of defect.

In short:

- The F1-scores of the different approaches on the test set range from 0.387 to 0.507.
- The best scoring reconstruction-based method is the Adversarial SSIM-AE and the best scoring feature representation-based method the PaDiM-WR50. On both rail and end post crop test sets, both PaDiM variants still perform better than all AE variants.
- On rail crop images the A_3 maximum patch function scores highest overall, while on end post crop images the A_2 standard deviation function works better.
- The approaches score on average higher on end post crop images than on rail crop images.
- The highest ranking approach is the PaDiM-WR50 with A_2 scoring on end post crops with a F1-score of 0.507. The accompanying recall is 0.755 and precision is 0.382.
- By categorizing defective IRJs by type of defect, this approach is shown to retrieve nearly all IRJs with spark erosion and squats. It struggles with IRJs showing end post constriction, crumble or play.

7 Discussion and conclusions

This chapter discusses the results of the experiments and highlights a few limitations. Furthermore, it concludes the thesis by answering the research questions and by providing recommendations.

7.1 Discussion

This section discusses the most important outcomes presented in Section 6.2 and attempts to explain these on the basis of the localization maps created by each method. The localization maps of example input images of functional and defective IRJs are shown in Figures 26-29 at the end of this chapter. In these maps, blue pixels correspond to low abnormality, green and yellow pixels to moderate abnormality and red pixels to high abnormality.

Mediocre detection performance. The calibration and test results in Sections 6.2.2 and 6.2.3 show that the best performing approaches are capable of creating reasonable separation between the anomaly scores of functional and defective IRJs classes. However, the AUC scores between 0.506 and 0.746 during calibration are lower than the scores seen with similar methods on the MVTec AD dataset, as presented in Section 3.3.3. After finding suitable thresholds for each approach, the F1-scores between 0.387 and 0.507 are also slightly worse than expected. The mediocre performance is mainly explained by the difficult IRJ imagery, given that the same methods perform significantly better on the MVTec AD data. The localization maps should highlight what exactly makes IRJ imagery difficult.

Firstly, the localization maps show that both reconstruction-based and embedding-based approaches are very sensitive to small harmless irregularities in the rail surface. This sensitivity is inherent to a semi-supervised detection approach and allows for the detection of multiple types of defects. However, in this case it works against itself as metal rail surfaces are inherently irregular and these irregularities typically have no correlation with an IRJ being functional or defective. Irregularities can also not be modelled as the methods can only capture the most frequently observed patterns. This behaviour is for instance seen in the reconstructions of the AE methods, which are all slightly blurry. In the end, these irregularities lead to relatively high anomaly scores for functional IRJs and a substantial overlap with the anomaly scores of defective IRJs.

Secondly, it is shown after testing that the highest scoring approach is more often able to retrieve IRJs with spark erosion region or squats than IRJs with end post defects (*Constriction*, *Play* and *Crumble*). Figures 31-40 in Appendix A.4 shows localization maps per defect sub-label. It is observed that localization is more accurate in images of IRJs with spark erosion or squats. This is probably explained by their larger size and stronger deviation from regular rail surface textures. In images of IRJs with end post defects the methods are indecisive in terms of localization as the actual defects are very small and because of the previously mentioned surface irregularities.

Lastly, it is observed that imperfect image cropping also leads to detection errors. As explained in Section 4.3, the heuristics for rail and end post cropping are not always flawless, resulting in near-rail environment being visible in a small number of input images. These areas are irregular and thus highlighted in the localization maps by the semi-supervised methods. A handful of functional IRJs are therefore falsely classified as defective IRJs, such as the example in Figure 25.

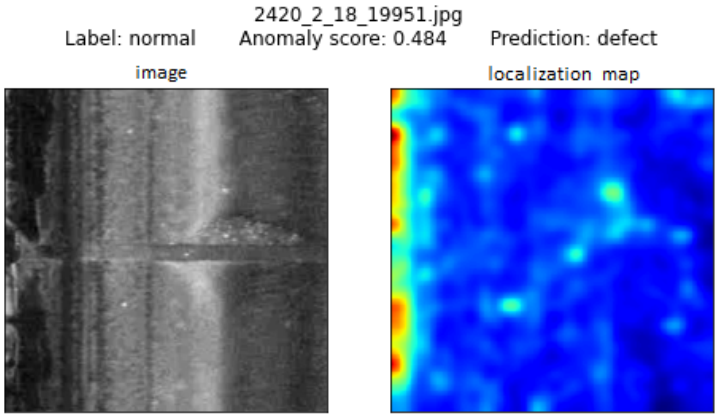


Figure 25: This functional IRJ is falsely classified as defective due to imperfect end post region cropping.

Performance difference between methods. The test results furthermore show that the feature representation-based methods are significantly better than the reconstruction-based methods for detection of defective IRJs. This aligns with the results on the MVTec AD data and can only be explained by the intrinsically different modelling setup. An actual theoretical explanation is difficult to provide as all methods are based on CNNs, which are known for being hard to interpret. Within the group of embedding-based models, the PaDiM-WR50 probably has better detection performance than the PaDiM-R18 as the Wide ResNet-50 variant extracts more features. For the reconstruction models, the Adversarial SSIM-AE arguably performs better than the SSIM-AE because of the adversarial training component. Both methods perform better than the L2-AE as the SSIM-loss is more sensitive to small differences in contrast and texture.

Relation between image crops and scoring functions. In addition, the test results show that A_3 maximum patch scoring is best combined with the rail cropped images and A_2 standard deviation scoring with the end post cropped images. From Figures 26 and 27 it can be observed that with rail crops the abnormality is typically found in a condensed area. Apart from the fact if this area is rightly or wrongly linked to a defect, it aligns best with the assumption of maximum patch scoring that an IRJ can already be defective if it shows high abnormality in a specific image region, as explained in Section 5.4. Then Figures 28 and 29 show that the more close-up view provided by the end post crops leads to a higher sensitivity to small imperfections and thus more spread in the highlighted abnormalities. This aligns best with with A_1 mean scoring and A_2 standard deviation scoring as these techniques consider the entire localization map. In these specific localization maps, functional IRJs generally show an evenly distributed moderate abnormality (green highlighting) as no image region stands out. The maps of defective IRJs on the contrary show more defined abnormality in terms of more blue and red highlighting. This explains why standard deviation scoring works better than mean scoring for end post cropped images.

Unexpected behaviour of PaDiM-R18. Inspection of the localization maps lastly reveals that the PaDiM-R18 model on the rail crop data exhibits unexpected behaviour. It exclusively highlights anomalous areas in the image corners, which in fact are part of the image padding used to make the rail crops square, as explained in Section 4.3. Localization of this PaDiM-R18 model is thus entirely incorrect. It is unknown what causes this behaviour as the PaDiM-WR50 does not display it on the same data. Nevertheless, this study will

disregard the results of the three approaches using this method. Interestingly, one of these approaches showed the highest F1-score of 0.502 on the rail data.

7.2 Limitations

The presented research has two limitations both in data processing and in modelling. The first data processing limitation was already reflected on in Section 7.1 and concerns the sometimes imperfect image cropping. The presence of near-rail environment in cropped images can lead to unintended false positives on functional IRJs, negatively impacting overall detection performance. The second data related limitation is that the sub-labelling of our dataset is not completely free of errors as discussed in Section 4.2. For instance, vertical indentation of IRJs was seen in some images during labelling but often placed under the *Uncertain* label as its severity could not accurately be examined from the top-down camera angle. Other cases of indentation were labeled as *Constriction* or *Squat*. Although, this has no major consequences for the main labels (*Functional* and *Defect*) and the overall detection performance of our methods, it should be kept in mind when observing the defect specific detection performance in Section 6.2.3 or when using the dataset for future research.

The first limitation in terms of modeling is that the PaDiM-WR50 approach for rail cropped images could only be trained on 1000 out of 1907 training samples. As explained in Section 6.2.1, the hardware encountered crashes when attempting to train on more samples. The smaller training set size arguably affects the detection performance of the PaDiM-WR50 implementation negatively, although it still performed better than the other methods on the test set. The second modelling related shortcoming is that the PaDiM-R18 approach for rail cropped images falsely identifies abnormality in the image padding as explained in Section 7.1. This padding was added to meet the requirement of the ResNet architectures for square image inputs. Although the exact reason for this behaviour is unknown, the results of these specific approaches should be disregarded.

7.3 Conclusions

The aim of this research was to examine the possibility to detect defective IRJs with a semi-supervised learning-based approach, as stated in Section 1.2. This approach exclusively uses images of functional IRJs to model normality and assumes that everything deviating from this normality is considered abnormal. After having reviewed related works and having implemented various semi-supervised detection methods, we are now able to formulate answers to the three research sub-questions and subsequently the main question.

1.*How to express abnormality of an IRJ with semi-supervised learning methods?*

Two distinct categories of semi-supervised learning methods are identified, which are both suitable for expressing the abnormality of IRJs in images. The first category of methods is based on image reconstruction ability. Convolutional AE or GAN architectures are trained for reconstruction of functional IRJ images, based on the assumption it makes them unable to afterwards reconstruct the defects present in defective IRJ images. An image of a damaged IRJ will be reconstructed as the same IRJ without defects. The pixel-wise differences between input and reconstruction result in a localization map showing the local abnormality of the IRJ. This study implemented this category of methods with the L2-AE, SSIM-AE and Adversarial SSIM-AE models, which architectures are based on GANomaly of [27].

The second category of methods is based on feature representation extractions of CNNs, which are typically pre-trained for ImageNet classification. The methods each apply hand-

crafted techniques to build a model of normality from the extracted embeddings of functional IRJ images, based on the assumption that embeddings of functional and defective IRJs deviate. This study implemented the PaDiM framework by [1], which models the embedding vectors of functional IRJ image patches as multivariate Gaussian distributions. The local abnormality of a defective IRJ image is then highlighted in a localization map of Mahalanobis distances between the embedding vectors and modelled distributions.

The resulting localization maps are aggregated to global scores, depicting the overall abnormality of IRJs. This study opted to perform aggregation with mean, standard deviation and maximum patch scoring techniques. These are based on the assumptions that localization maps of defective IRJs show either higher abnormality or more spread of abnormality than the ones of functional IRJs.

2. *Which detection approach is most promising in terms of test set performance?*

Several semi-supervised detection approaches, which differ in data processing, localization method and scoring function, have the ability to distinguish functional from defective IRJs to some extent. According to the results on the test set in Section 6.2.3, the most promising approach is the PaDiM-WR50 method which uses A_2 standard deviation scoring and is trained on end post cropped images. This specific approach is able to create the largest separation between the anomaly scores of functional and defective IRJs. Given the optimal threshold of 0.232, the performance on the test set is expressed with a F1-score of 0.507, recall of 0.755 and precision of 0.382. This approach thus retrieves over 75% of all defective IRJs, while roughly 60% of the detections are incorrectly made on functional IRJs. The correct detection do include over 88% of IRJs with spark erosion and over 95% of IRJs with squats. The IRJs with end post specific defects are only retrieved between 50 and 65% of the time.

3. *Which IRJ image crop is the most promising in terms of test set performance?*

The detection methods perform better on the end post cropped images than on the rail cropped images as the aggregated F1-scores compare with 0.463 to 0.454, respectively. This difference of approximately 2% indicates that it is slightly more beneficial to crop the IRJ images such that the end post region is centrally in frame. It also proves that it is important to keep imagery as uniform as possible for the effectiveness of a semi-supervised detection approach. The only downside of end post region cropping is that 1769 of the 4252 images in the dataset could not be used due to the current division of raw video inspection data into separate frames, as explained in Section 4.4.

Given these conclusions, the main research question can now be answered.

How and to what extent can a semi-supervised learning-based approach be applied to detect IRJs with various defects?

This thesis demonstrates the feasibility of using a semi-supervised learning-based approach for the detection of defective IRJs. This approach overcomes the challenge of detecting IRJs with defects that can have varying appearances and effectively handles the class imbalance present in the manually labelled dataset of over 4250 images of functional and defective IRJs. The research furthermore fills a gap in existing literature on railway defect detection.

The research evaluated three reconstruction-based methods (L2-AE, SSIM-AE, and Adversarial SSIM AE) and two feature representation-based methods (PaDiM-R18 and PaDiM-

WR50), which were trained using images of only functional IRJs to establish normality. Each method is able to afterwards generate localization maps that indicate local IRJ abnormality and can be scored using three different scoring functions (mean, standard deviation, and maximum patch). The impact of IRJ image processing was also studied through rail surface cropping and end post region cropping.

The best performing approach is the PaDiM-WR50 method using the A_2 standard deviation scoring on end post region cropped images, yielding a recall of 0.755, precision of 0.382, and an F1 score of 0.507 on the test set. The method is effective in identifying defective IRJs, particularly those with spark erosion regions and squats, although it also produces many false detections on functional IRJs. The PaDiM-WR50 method also has reasonable results on rail surface cropped images using the A_3 maximum patch scoring function, with an F1 score of 0.498. Other approaches, including those incorporating different scoring functions and reconstruction-based methods, performed worse and in some cases were only slightly better than random prediction.

The precarious performance of the semi-supervised methods can be attributed to the challenging nature of IRJ images as identical methods have shown great results in literature. Primarily the irregular and harmless imperfections in the metal rail surface of functional IRJs and the subtlety of most defects in defective IRJs make that both classes are sometimes difficult to distinguish. As discussed in Section 4.2, even domain experts sometimes disagree on the classification of defective IRJs, indicating that perfect detection does not exist. While the semi-supervised approach shows potential, it may not yet be suitable as a standalone solution for detecting defective IRJs.

7.4 Recommendations

The results of this study demonstrate that certain semi-supervised approaches can differentiate between functional and defective IRJs with adequate accuracy, however, the performance is not entirely sufficient for practical use by ProRail yet. Hence, this section provides recommendations for further research.

The first recommendation is to connect track circuit failure data with corresponding IRJ images. This study classifies IRJs as defective according to criteria set by ProRail, but it is unknown which of these IRJs are actually causing track circuit failures. Knowledge about which IRJs are truly defective could potentially result in improved model performance as these IRJs may show more severe defects and would therefore be more easily distinguishable from undamaged IRJs.

The second suggestion is to implement the PatchCore framework of [2] instead of the PaDiM model as the feature representation-based method. As mentioned in Section 3.3.3, PatchCore outperforms PaDiM on the MvTecAD benchmark with a state-of-the-art AUC of 0.991, compared to PaDiM's 0.953. It would be worth exploring if a similar performance improvement can be achieved for detecting defective IRJs. Combined with the additional connection between track circuit failures and defective IRJs, this setup could potentially work to standards. PatchCore was not implemented in this research because its image scoring uses a KNN algorithm and was therefore not comparable to our relatively simple scoring functions.

The third recommendation may serve as an alternative when the previous two recommendations do still not result in satisfactory detection performance. It is to switch to a supervised detection approach similar to the literature in Section 3.2. The number of damaged IRJs in the data, as observed in Section 4.2, seems substantial enough to train separate supervised

models for each type of defect. It is then recommended to use segmentation models because these only use defect-specific information for classification, which can improve performance on the smaller end post defects. Most methods used by the AMI Renewal department are supervised segmentation methods, such as the railway fastener defect detection solution. The convolutional U-Net architecture by [32] may be a good starting point.

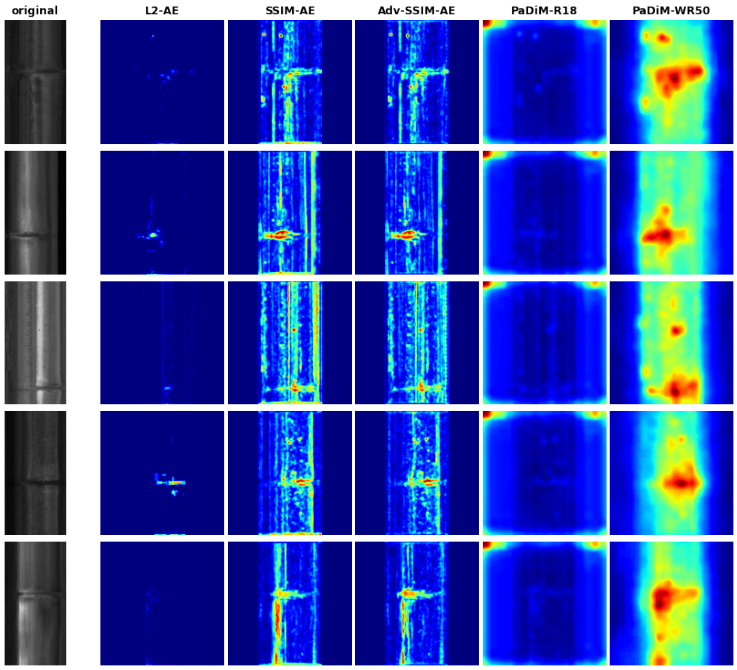


Figure 26: Rail cropped test samples (*Functional*) with corresponding localization maps

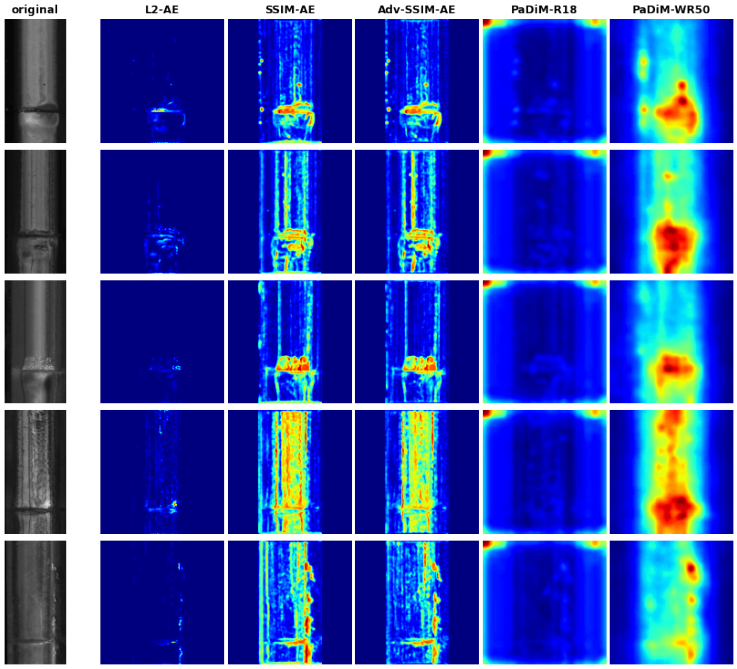


Figure 27: Rail cropped test samples (*Defect*) with corresponding localization maps

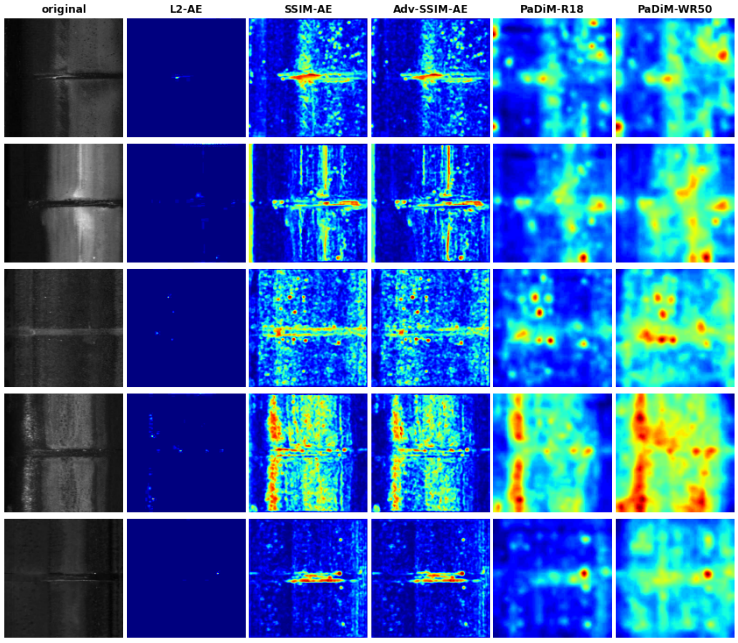


Figure 28: End post cropped test samples (*Functional*) with corresponding localization maps

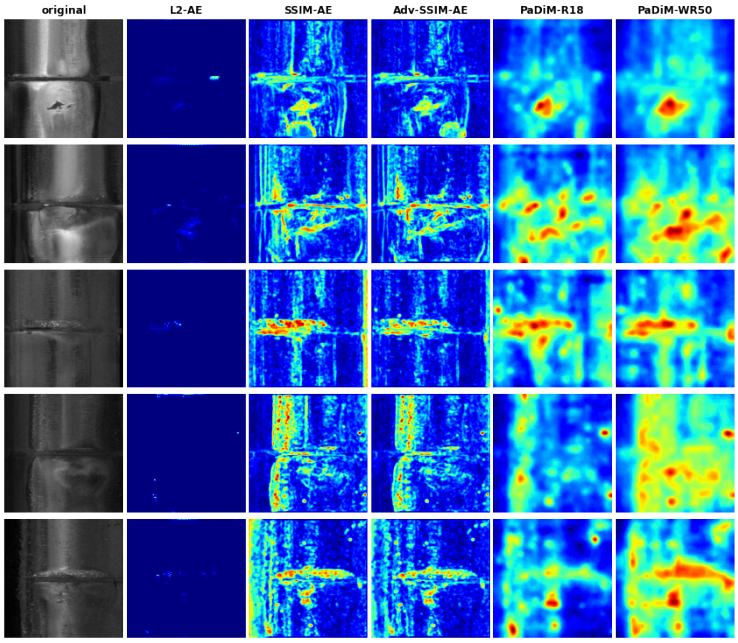


Figure 29: End post cropped test samples (*Defect*) with corresponding localization maps

References

[1] T. Defard, A. Setkov, A. Loesch, and R. Audigier, "Padim: a patch distribution modeling framework for anomaly detection and localization," in *International Conference on Pattern Recognition*, pp. 475–489, Springer, 2021.

[2] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, and P. Gehler, "Towards total recall in industrial anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14318–14328, 2022.

[3] Q. Li, Z. Zhong, Z. Liang, and Y. Liang, "Rail inspection meets big data: Methods and trends," in *2015 18th International Conference on Network-Based Information Systems*, pp. 302–308, IEEE, 2015.

[4] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.

[5] Q. Li and S. Ren, "A visual detection system for rail surface defects," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1531–1542, 2012.

[6] H. Zhang, X. Jin, Q. J. Wu, Y. Wang, Z. He, and Y. Yang, "Automatic visual detection system of railway surface defects with curvature filter and improved gaussian mixture model," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 7, pp. 1593–1608, 2018.

[7] V. Vijaykumar and S. Sangamithirai, "Rail defect detection using gabor filters with texture analysis," in *2015 3rd International Conference on Signal Processing, Communication and Networking (ICSCN)*, pp. 1–6, IEEE, 2015.

[8] H. Yu, Q. Li, Y. Tan, J. Gan, J. Wang, Y.-a. Geng, and L. Jia, "A coarse-to-fine model for rail surface defect detection," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 3, pp. 656–666, 2018.

[9] C. Mandriota, M. Nitti, N. Ancona, E. Stella, and A. Distante, "Filter-based feature selection for rail defect detection," *Machine Vision and Applications*, vol. 15, no. 4, pp. 179–185, 2004.

[10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[11] J. Yang, R. Xu, Z. Qi, and Y. Shi, "Visual anomaly detection for images: A survey," *ArXiv*, vol. abs/2109.13157, 2021.

[12] S. Faghih-Roohi, S. Hajizadeh, A. Núñez, R. Babuska, and B. De Schutter, "Deep convolutional neural networks for detection of rail surface defects," in *2016 International joint conference on neural networks (IJCNN)*, pp. 2584–2589, IEEE, 2016.

[13] Z. Liang, H. Zhang, L. Liu, Z. He, and K. Zheng, "Defect detection of rail surface with deep convolutional neural networks," in *2018 13th World Congress on Intelligent Control and Automation (WCICA)*, pp. 1317–1322, IEEE, 2018.

[14] J. H. Feng, H. Yuan, Y. Q. Hu, J. Lin, S. W. Liu, and X. Luo, "Research on deep learning method for rail surface defect detection," *IET Electrical Systems in Transportation*, vol. 10, no. 4, pp. 436–442, 2020.

[15] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998.

[16] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[17] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, "Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9592–9600, 2019.

[18] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[19] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *MLSDA'14*, 2014.

[20] D. Gong, L. Liu, V. Le, B. Saha, M. R. Mansour, S. Venkatesh, and A. v. d. Hengel, "Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1705–1714, 2019.

[21] S. Mei, H. Yang, and Z. Yin, "An unsupervised-learning-based approach for automated defect inspection on textured surfaces," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 6, pp. 1266–1277, 2018.

[22] C. Baur, B. Wiestler, S. Albarqouni, and N. Navab, "Deep autoencoding models for unsupervised anomaly segmentation in brain mr images," in *International MICCAI brainlesion workshop*, pp. 161–169, Springer, 2018.

[23] P. Bergmann, S. Löwe, M. Fauser, D. Sattlegger, and C. Steger, "Improving unsupervised defect segmentation by applying structural similarity to autoencoders," *arXiv preprint arXiv:1807.02011*, 2018.

[24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2014.

[25] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *International conference on information processing in medical imaging*, pp. 146–157, Springer, 2017.

[26] H. Zenati, C. S. Foo, B. Lecouat, G. Manek, and V. R. Chandrasekhar, "Efficient gan-based anomaly detection," *arXiv preprint arXiv:1802.06222*, 2018.

[27] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "Ganomaly: Semi-supervised anomaly detection via adversarial training," in *Asian conference on computer vision*, pp. 622–637, Springer, 2018.

[28] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.

[29] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[30] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," *IEEE Transactions on computational imaging*, vol. 3, no. 1, pp. 47–57, 2016.

[31] S. Akçay, A. Atapour-Abarghouei, and T. P. Breckon, "Skip-ganomaly: Skip connected and adversarially trained encoder-decoder anomaly detection," in *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, 2019.

[32] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.

[33] A.-S. Collin and C. De Vleeschouwer, "Improved anomaly detection by training an autoencoder with skip connections on images corrupted with stain-shaped noise," in *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 7915–7922, IEEE, 2021.

[34] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[35] O. Rippel, P. Mertens, and D. Merhof, "Modeling the distribution of normal data in pre-trained deep features for anomaly detection," in *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 6726–6733, IEEE, 2021.

[36] N. Cohen and Y. Hoshen, "Sub-image anomaly detection with deep pyramid correspondences," *arXiv preprint arXiv:2005.02357*, 2020.

[37] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*, pp. 6105–6114, PMLR, 2019.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

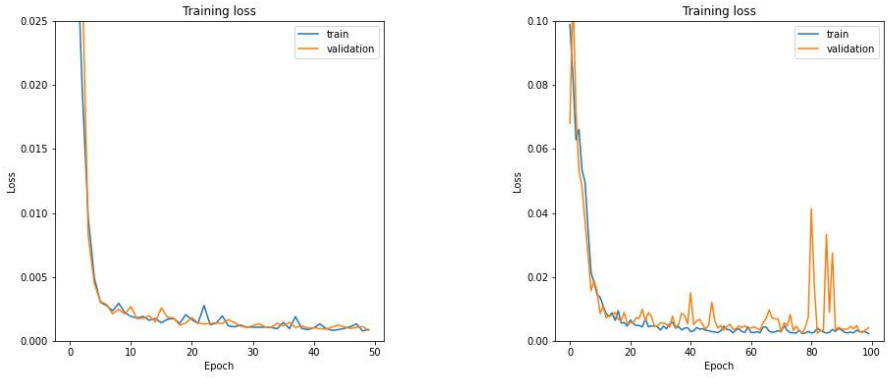
[39] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[40] A. F. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever, "Multiscale vessel enhancement filtering," in *International conference on medical image computing and computer-assisted intervention*, pp. 130–137, Springer, 1998.

[41] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

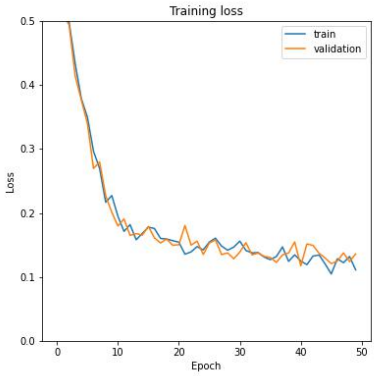
A Appendix

A.1 Training loss curves of reconstruction-based methods

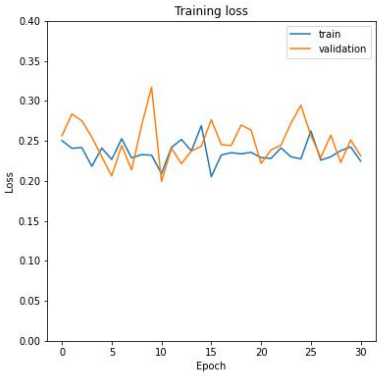


(a) L2-AE (rail)

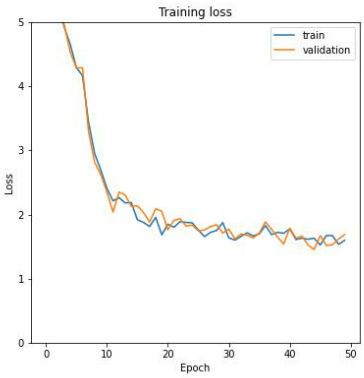
(b) L2-AE (IRJ)



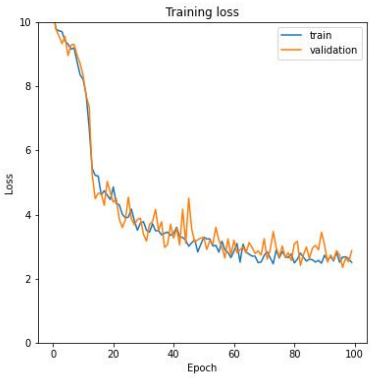
(c) SSIM-AE (rail)



(d) SSIM-AE (IRJ)



(e) Adversarial SSIM-AE (rail)



(f) Adversarial SSIM-AE (IRJ)

Figure 30: Loss curves of reconstruction algorithms after training on rail data (a,c,e) and IRJ data (b,d,f)

A.2 AUC scores on calibration set

	L2-AE	SSIM-AE	Adv. SSIM-AE	PaDiM-R18	PaDiM-WR50
A_1	0.566	0.525	0.527	0.606	0.593
A_2	0.582	0.551	0.560	0.680	0.667
A_3	0.612	0.597	0.624	0.714	0.731

Table 14: AUC scores of each model scoring combination on the rail calibration 0.set

	L2-AE	SSIM-AE	Adv. SSIM-AE	PaDiM-R18	PaDiM-WR50
A_1	0.528	0.526	0.506	0.555	0.649
A_2	0.512	0.578	0.601	0.708	0.746
A_3	0.551	0.554	0.568	0.622	0.664

Table 15: AUC scores of each model scoring combination on the IRJ calibration set

A.3 Precision and recall on test set

	L2-AE	SSIM-AE	Adv. SSIM-AE	PaDiM-R18	PaDiM-WR50
A_1	0.311 / 0.654	0.285 / 0.982	0.290 / 0.921	0.296 / 0.807	0.284 / 0.917
A_2	0.314 / 0.649	0.302 / 0.746	0.299 / 0.754	0.322 / 0.851	0.332 / 0.864
A_3	0.335 / 0.789	0.315 / 0.855	0.361 / 0.732	0.373 / 0.768	0.358 / 0.820

Table 16: Precision / recall scores of each model scoring combination on the rail test set

	L2-AE	SSIM-AE	Adv. SSIM-AE	PaDiM-R18	PaDiM-WR50
A_1	0.311 / 0.683	0.302 / 0.971	0.287 / 0.986	0.305 / 0.942	0.323 / 0.878
A_2	0.320 / 0.906	0.318 / 0.835	0.308 / 0.899	0.384 / 0.691	0.382 / 0.755
A_3	0.309 / 0.518	0.297 / 0.920	0.308 / 0.827	0.338 / 0.849	0.340 / 0.842

Table 17: Precision / recall scores of each model scoring combination on the IRJ test set

A.4 Localization maps per defect sub-label

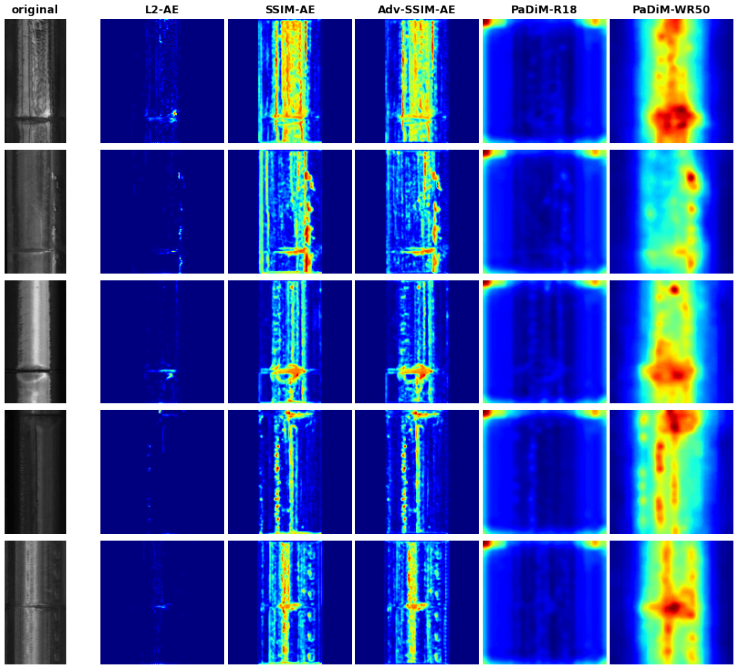


Figure 31: Rail cropped test samples (*Constriction*) with corresponding localization maps

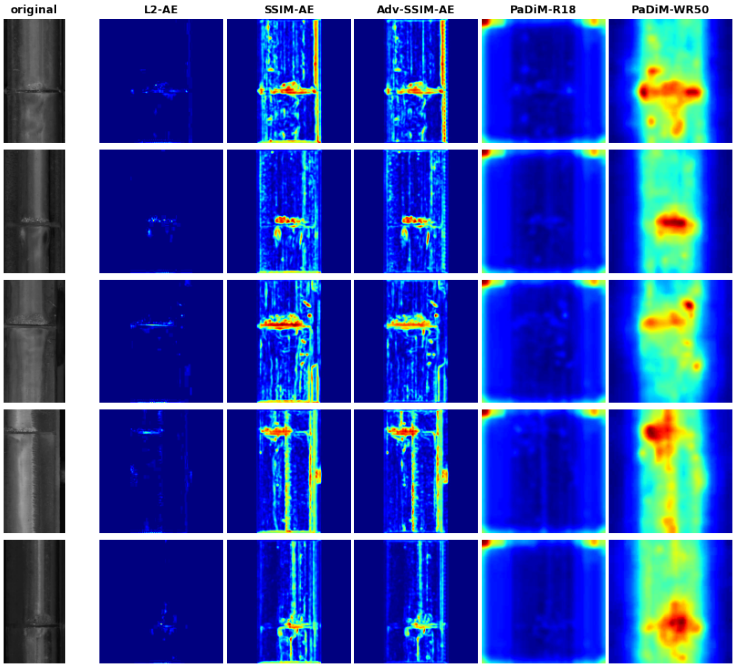


Figure 32: Rail cropped test samples (*Spark erosion*) with corresponding localization maps

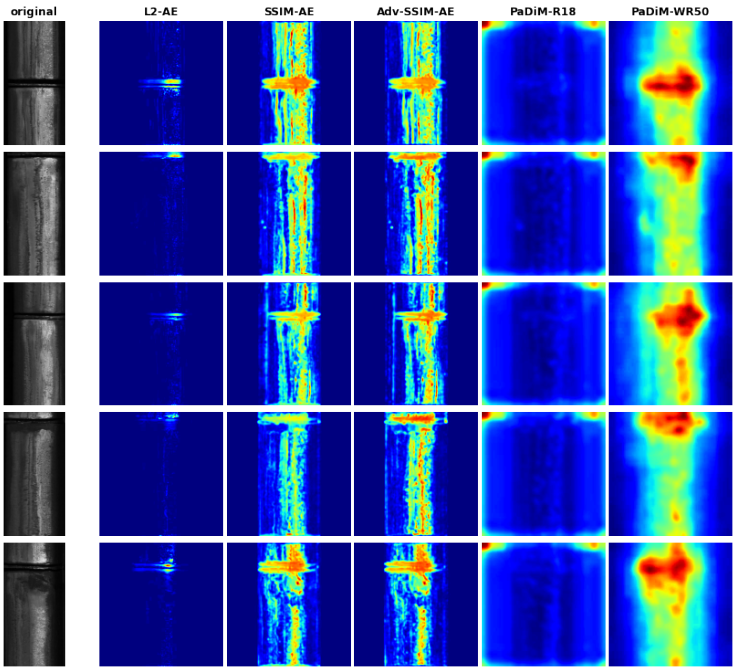


Figure 33: Rail cropped test samples (*Play*) with corresponding localization maps

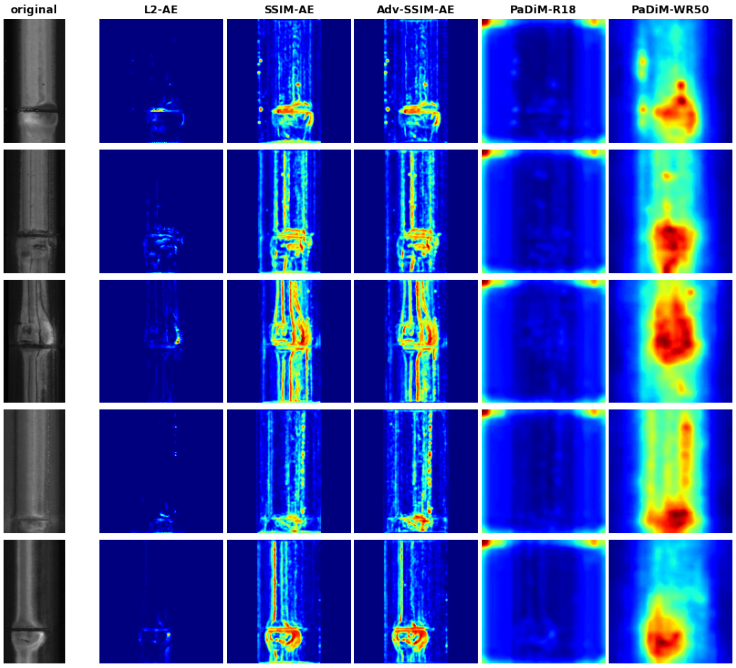


Figure 34: Rail cropped test samples (*Squat*) with corresponding localization maps

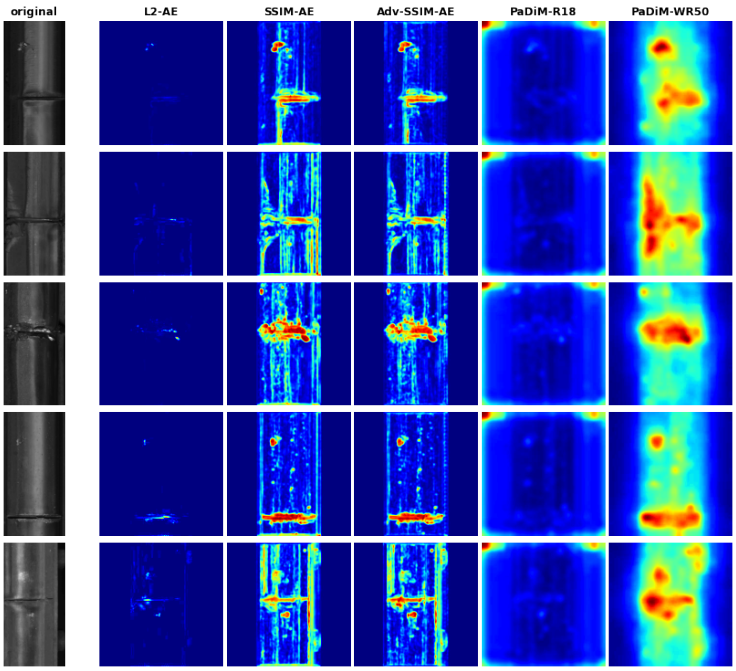


Figure 35: Rail cropped test samples (*Crumbled*) with corresponding localization maps

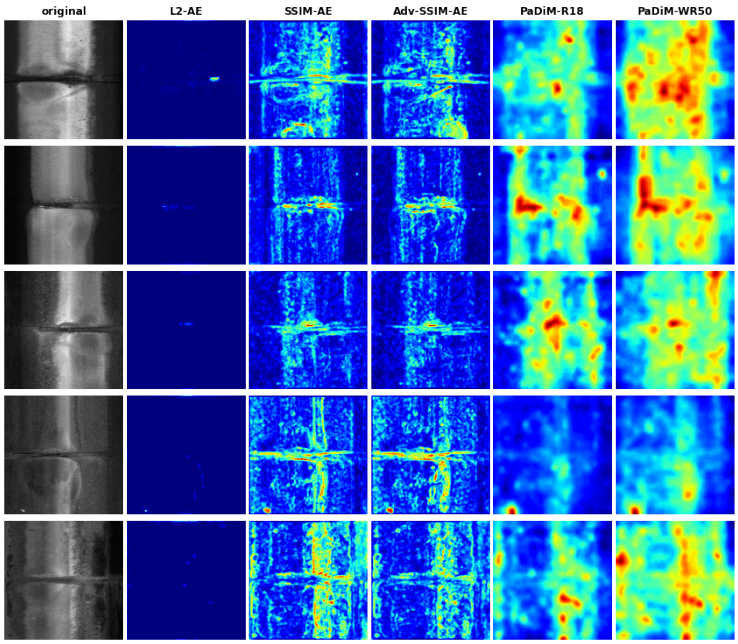


Figure 36: End post cropped test samples (*Constriction*) with corresponding localization maps

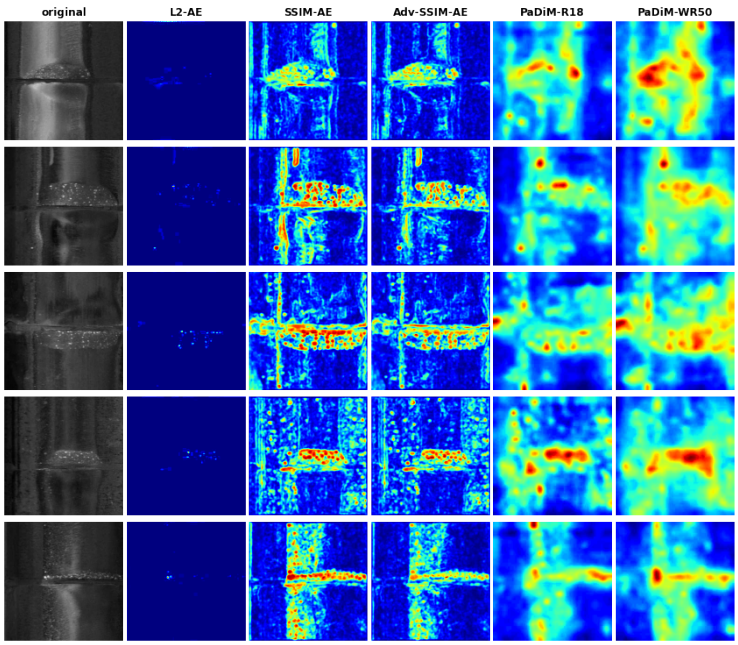


Figure 37: End post cropped test samples (*Spark Erosion*) with corresponding localization maps

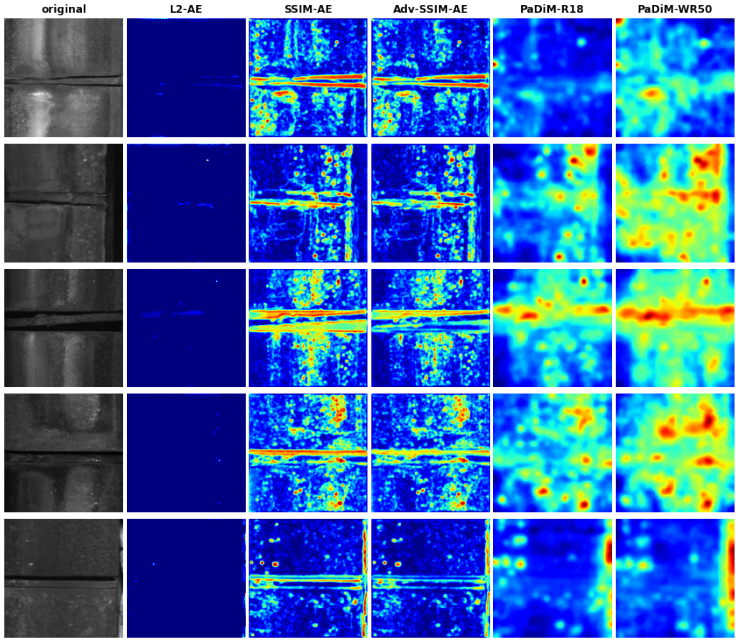


Figure 38: End post cropped test samples (*Play*) with corresponding localization maps

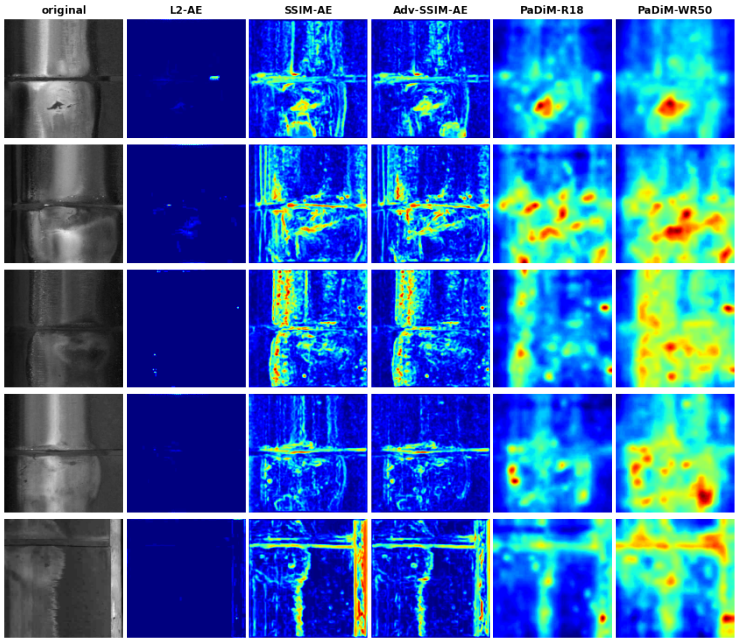


Figure 39: End post cropped test samples (*Squat*) with corresponding localization maps

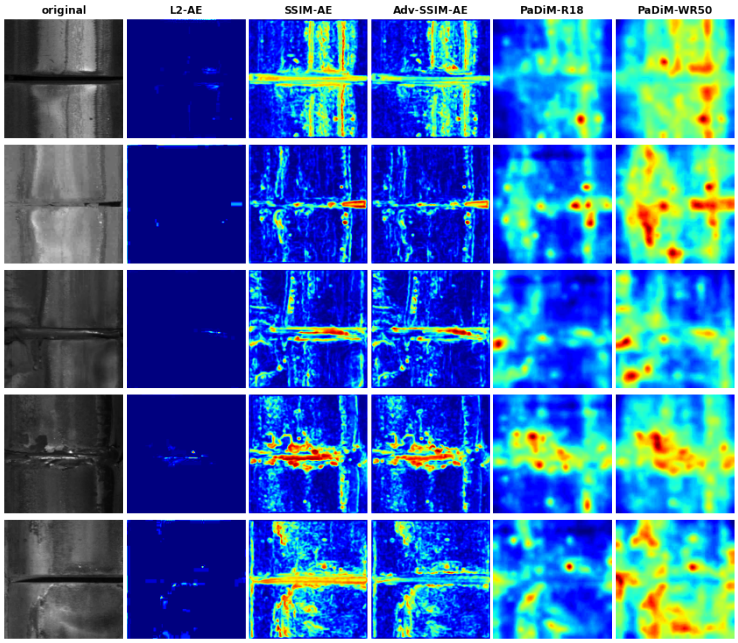


Figure 40: End post cropped test samples (*Crumbled*) with corresponding localization maps