Proceedings of the Sixth International Conference on
Railway Technology: Research, Development and Maintenance
Edited by: J. Pombo
Civil-Comp Conferences, Volume 7, Paper 8.4
Civil-Comp Press, Edinburgh, United Kingdom, 2024
ISSN: 2753-3239, doi: 10.4203/ccc.7.8.4

Automating Multi-Analytical Tasks in Machine-Vision Enabled Rail Surface Inspections: A Three-Stage Deep Learning Based Method

T. Wang

Institute of Rail Transit, Tongji University
Shanghai, China

Abstract

Main tasks in railway inspection includes identifying, locating and classifying rail surface defects. In this paper, we develop a three-stage deep learning based framework for automating multiple analytical tasks in the rail surface defect inspection via using rail images. It is capable of identifying the presence, the sizes with coordinates, and the category of various defects. The first stage employs an autoencoder based generative model to identify the input image containing defects, which then trigger the second stage with a segmentation model for locating defects at the pixel level. Finally, the defect category is classified with segmented and cropped defect regions obtained from the previous stages. The proposed method is evaluated thoroughly with different performance criteria on a real dataset. Moreover, due to limited publicly available datasets in railway inspection, we synthesized a new dataset to further verify the generalizability of the proposed framework. Results of the computational studies validated its accuracy and suitability on a self-powered inspection system in terms of the computational load, performance, and inference time.

Keywords: deep learning, defect inspection, image analytics, multi-task learning, rail transport safety, system health management.

1 Introduction

The rail surface defect detection (RSDD) is critical to ensure the safety of rail transport. Traditional RSDD is performed manually or based on processing signals [1, 2]. They target more on the diagnosis tasks rather than automation of manual inspection routines. Meanwhile, sensors, such as the stereo cameras, radar, and LIDAR, are prohibitively expensive and require considerable power during the data collection, which might limit the ease-of-the usage.

Technologies for collecting visual data and processing images have been advanced, and the early vision-based RSDD methods mainly rely on the texture characteristics of the image [3, 4]. The recent development of machine learning technologies enables the RSDD based on the automatic feature engineering [5]. Although these methods can yield reliable and effective results based on sufficient training samples, it is challenging to gather a large number of unhealthy rail images due to the rare occurrence and appearance variations of the surface defects. Even for the semi-supervised deep learning approaches [6], they only realize RSDD at the image level. It is hard to identify the location and size of the surface defects at the same time.

This paper proposes an automated rail image multi-analytics framework for the RSDD task. Given input images, the framework integrates three analytical functions, identification, segmentation, and classification. Figure 1 offers an example of testing results. The different types of surface defects are well detected with segmentation masks, bounding boxes, and categories, showing the potential to evolve the manual analytics of rail images to a machine-vision enabled automated process.

The main contributions of this work are summarized as follows:

1. We propose a novel framework for realizing multiple analytical functions in RSDD. Compared with previous methods focusing one particular analytical function in RSDD, the proposed framework synergizes three distinct stages to identify, localize, and classify surface defects based on rail images. To the best of our knowledge, the proposed framework is the first to simultaneously address multiple analytical tasks in the RSDD.
2. The necessity and value of each stage in the proposed framework is verified through a set of computational experiments, which form an ablation study. The developed framework can achieve high accuracies and performs robustly regarding multiple analytical tasks, representing a suitable solution for the defect inspection on the rail surface.
3. The applicability of the proposed framework is proved through extended experiments on a new dataset. The results show that the proposed framework has the generalization capability in applications to deal with various targets in the RSDD scenario.

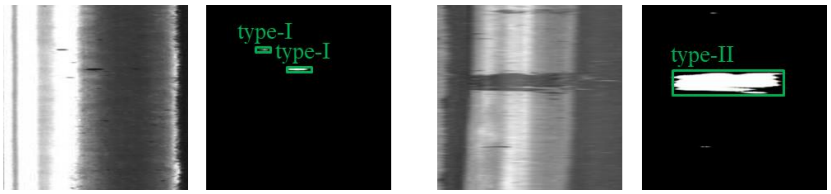


Figure 1: Test samples of different type of defects.

2 Related Work

The traditional practice of inspecting rail surface defects relies on experienced maintenance crews walking along rails to manually check the health condition of rails periodically, which is time-consuming and labor-intensive. With the innovation of sensing technologies, it becomes feasible to realize the defect detection by extracting features from collected signals, such as the acoustic emission [7] and vibration [8]. Nevertheless, the signal-based methods are often only suitable for the detection of specific defects. Their accuracies are also limited by the quality of the collected signal and the surrounding environment.

Meanwhile, the advancement of machine-vision techniques effectively addresses many real-world applications, such as the frontal obstacle detection [9], railway track extraction [10], as well as the problem of RSDD. In terms of different stages of the technological evolution, previous works relating to the vision-based RSDD can be categorized into the traditional image processing and machine learning methods. The former relies on the explicitly engineered features based on texture analyses, while the latter serves transforming the image data into complex, abstract, and learnable representations. The RSDD problem based on traditional methods was essentially a texture analysis problem, which could be solved by the edge detection [3], filtering [11, 12], and thresholding [13, 14]. Although they focus on the explicitly engineered features in RSDD, it can be challenging in complex cases with different illuminations or backgrounds. The rich accumulation of traditional methods and the rapid development of learning principles resulted in the evolution of machine learning approaches to overcome the requirement of more complex features for a specific defect.

The machine learning methods can be classified into two groups based on their training mechanisms, the supervised and semi-supervised methods. The former type of methods predefined the detectable features and was trained under supervision. Typical methods include the support vector machine (SVM) [15, 16], k-nearest neighbor (KNN) [12], and deep convolutional neural networks (DCNNs) [17-21]. Although the supervised methods have achieved a reasonable performance on RSDD task, it is impossible to collect and label sufficient defect samples due to the limitation and randomness of the defects in realistic cases, which makes the final detection and classification result unstable. The semi-supervised approaches are beneficial when addressing the lack of defective image data collected in the railway context for RSDD, such as the autoencoder (AE) [22] and generative adversarial network (GAN) [6].

In general, we notice that most vision-based RSDD models are trained in a supervised manner, requiring sufficient training samples from existing classes. Few of the semi-supervised methods aim to simultaneously identify and localize the defects in the rail images from both image and pixel levels. There is an urgent need to introduce more reliable and effective methods on RSDD. Thus, we put forward a novel framework to integrate the identification, segmentation, and classification of rail surface defects.

3 Method Description

The proposed RSDD framework consists of three sequential stages enabling multiple analytical functions as shown in Figure 2. In this section, we explain each stage in terms of its task, architecture, training procedure, and testing procedure.

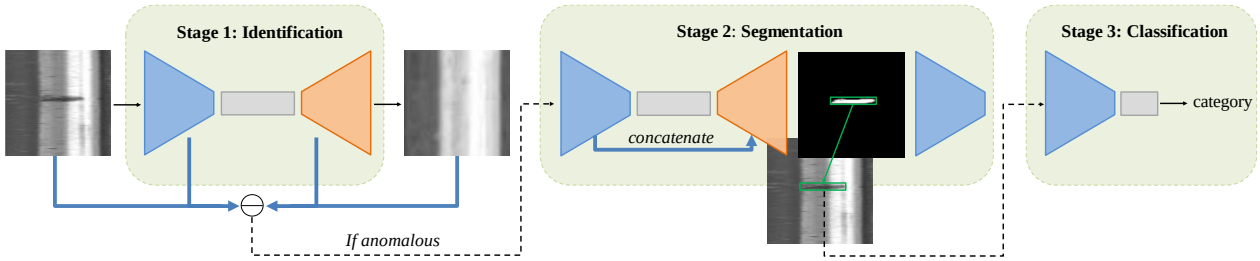


Figure 2: Overall view of the proposed three-stage method.

3.1 Identification of Unhealthy Rails

Taking inspirations from [23], the first stage serves the identification of the unhealthy rail (IUR). As illustrated in Figure 3, given the rail image $x \in \mathbb{R}^{C\times H\times W}$ as input, the IUR based on an AE learns to reconstruct the healthy version of x as follows:

$$\hat{x} = G_\phi(z) = G_\phi(E_\theta(x)), \tag{1}$$

where E_θ is an encoder network and G_ϕ is a decoder network, and $z \in \mathbb{R}^d$ is the bottleneck features in IUR as well as a lower-dimensional manifold of x. In training process of IUR, the dissimilarity between x and its reconstruction $\hat{x}$ is expressed via a match function $M(x, \hat{x})$. A high value of $M(x, \hat{x})$ indicates that x is unhealthy and contains surface defects.

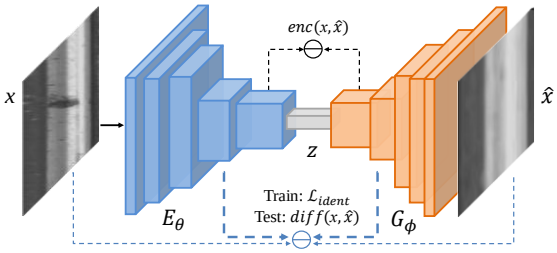


Figure 3: Architecture of the IUR.

The architecture of the E_θ in IUR is composed of 6 convolutional layers. The output number of channels in each layer is doubled starting from 16 and arriving to 512. The input images are resized to $128 \times 128 \times 3$ considering both computation burden and identification accuracy. With kernel size $k = 3$, stride $s = 2$, and padding size $p = 1$, $z \in \mathbb{R}^{512}$ is a 1-dimentional vector as the least representations of input image x. The LeakyReLu with a slope 0.1 is employed as the activation function after a batch normalization (BatchNorm) in all convolutional layers of E_θ. The decoder network G_ϕ is a symmetrical structure of E_θ. The number of channels in the feature map is halved after each and reduced to 16 at the end of G_ϕ. To make the input x and its

reconstruction $\hat{x}$ lie closely on the manifold z, the final output of G_ϕ is still an image with size of $128 \times 128 \times 3$ through up-sampling. Different from E_θ, the ReLu activation is utilized in the intermediate layers in G_ϕ except for the output layer, which utilizes the tanh(·) as the activation function to introduce nonlinearity to the reconstructions.

The AE architecture in IUR is trained via a semi-supervised approach that receives only healthy images without foreign objects during training. As illustrated in Figure 3, the loss function $\mathcal{L}_{ident}$ consists of two parts, the perceptual loss $\mathcal{L}_{perceptual}$ and the pixel loss $\mathcal{L}_{pixel}$. The former term describes the L_2 distance between the encoded features of x and $\hat{x}$ as well as encourages the output image $\hat{x}$ to be perceptually similar to x. The second term $\mathcal{L}_{pixel}$ is the commonly used error term directly describing the pixel difference between x and $\hat{x}$. Overall, the loss function $\mathcal{L}_{ident}$ is defined as follows:

$$\mathcal{L}_{ident} = \sum \mathbb{E}_{x,\hat{x}} \left\| f_{i,E_\theta}(x) - f_{i,G_\phi}(\hat{x}) \right\|_2 + \mathbb{E}_{x,\hat{x}} \|x - \hat{x}\|_1, \quad (2)$$

where $f_i(\cdot)$ denotes the i^{th} activation layer from both E_θ and G_ϕ.

With the AE well trained, one possible design of the match function, $M(x, \hat{x})$, based on the encoded features is defined as follows to express the fit of $\hat{x}$ to x:

$$enc(x, \hat{x}) = \mathbb{E}_{x,\hat{x}} \left\| f_{E_\theta}(x) - f_{G_\phi}(\hat{x}) \right\|_1, \quad (3)$$

where $f(\cdot)$ denotes the last activation layer of E_θ and G_ϕ. The computation of $enc(x, \hat{x})$ directly derives from $\mathcal{L}_{ident}$. Since G_ϕ only reconstruct the healthy rails devoid of any surface defects, a threshold based on $enc(x, \hat{x})$ is required during testing. A rail image with $enc(x, \hat{x})$ larger than the threshold is identified as unhealthy, and the next stage is activated. A difference map $diff(x, \hat{x})$ is defined at the same time for detecting the location and the size of the defect in the next stage:

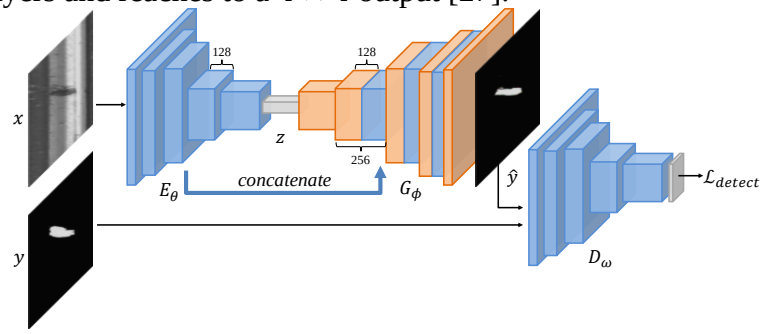
$$diff(x, \hat{x}) = \|x - \hat{x}\|_1 + \sum \left\| f_{i,E_\theta}(x) - f_{i,G_\phi}(\hat{x}) \right\|_2, \quad (4)$$

3.2 Segmentation of Defect Regions

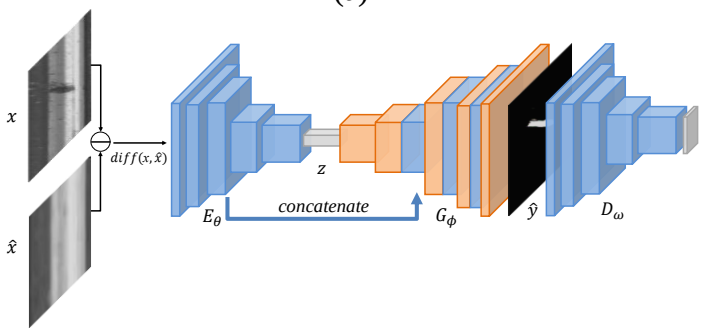
The segmentation of the defect region (SDR) aims to identify the location and size of the defect. Once the stage is activated, the stacked differences $diff(x, \hat{x})$ become the input of the SDR to generate the segmentation map $\hat{y}$. The SDR is required to have a mapping from the calculated difference $diff(x, \hat{x})$ to the ground truth label map y. Unlike other detection tasks that localize the objects with rectangular bounding boxes, the $\hat{y}$ can accurately reflect the shape of the defect with a pixel-wise segmentation mask as shown in Figure 1. Moreover, the $\hat{y}$ is utilized to crop the detected defects, which will be classified at the end of the framework.

The SDR follows an encoder-decoder-encoder architecture as in Figure 4. All layers in SDR utilize the form of convolution-BatchNorm-activation. Different from IUR, the G_ϕ in SDR adapts the network architecture of U-net [24] to skip z and directly concatenate all channels from corresponding layers in E_θ. For example, the feature

map with 128 channels in G_ϕ will concatenate the feature map of the same depth from E_θ to form a new feature map with 256 channels. Following typical convolutions, the depth of the stacked feature map is back to 128 and is ready for the next convolutional layer. This type of skip connection is formed in a symmetrical manner. Unlike generic U-net in segmentation tasks [25, 26], a discriminator network D_ω is added after the G_ϕ to enhance the capability for generating $\hat{y}$ as authentic as possible through an adversarial training. D_ω is also a replica of E_θ but without a BatchNorm to the first layer, It applies a dropout regularization with a probability $p = 0.2$ after LeakyReLu activation layers and reaches to a 4×4 output [27].



(a)



(b)

Figure 4: (a) Training procedure of SDR. (b) Testing procedure of SDR.

Supervised training is employed in the SDR. Given the rail image x, the segmentation map is generated through $\hat{y} = G_\phi(E_\theta(x))$. The ground truth y and the generated $\hat{y}$ are the targets to be discriminated by D_ω. The objective of the SDR training is based on the Wasserstein distance [28], which can be expressed as:

$$\mathcal{L}_{G,D} = \mathbb{E}_y[D_\omega(y)] - \mathbb{E}_z\left[D_\omega\left(G_\phi(z)\right)\right], \tag{5}$$

where G_ϕ and D_ω are trained simultaneously. The goal of G_ϕ is not only to fool D_ω but also generate $\hat{y}$ to be close to the ground truth y in L1 sense. Therefore, we explore the constraints directly based on y and $\hat{y}$ as in (6). The final objective $\mathcal{L}_{detect}$ is the sum of $\mathcal{L}_{G,D}$ and $\mathcal{L}_{E,G}$ and follows the two-player min-max game as expressed in (7):

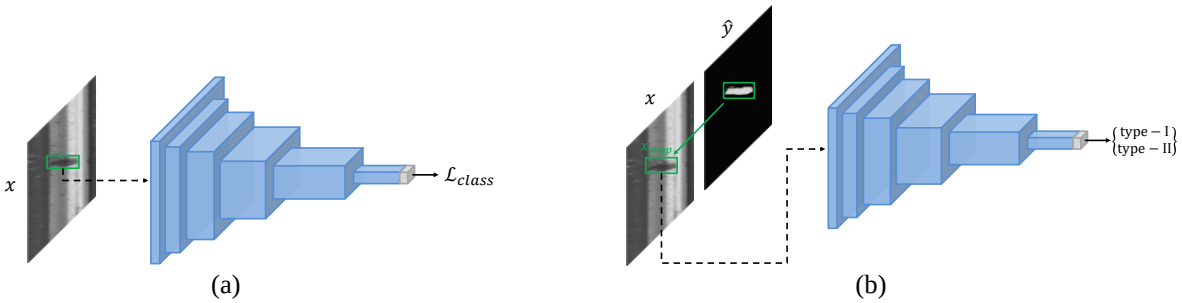
$$\mathcal{L}_{E,G} = \mathbb{E}_{x,y}\left\|y - G_\phi\left(E_\theta(x)\right)\right\|_1, \tag{6}$$

$$\arg\min_{G_\phi}\max_{D_\omega}\mathcal{L}_{detect} = \arg\min_{G_\phi}\max_{D_\omega}\mathcal{L}_{G,D} + \mathcal{L}_{E,G} \tag{7}$$

In testing, the segmentation map $\hat{y} = G_\phi(E_\theta(diff(x, \hat{x}))$ contains the highlighted defect region and the black background. Since the defect with a soft boundary is detected at the pixel level, it is easy to give a bounding box covering the whole defect region, which is then cropped from x and treated as the input of the next stage.

3.3 Classification of Defect Types

The classification of the defect type (CDT) is realized based on a simple CNN architecture as shown in Figure 5. The architecture of CDT owns a similar architecture as the encoder E_θ in IUR. There are additional two fully connected layers added after E_θ. One has 64 units, and another has two units $Z: \{z_1, z_2\}$. The input is the cropped defect region on the rail image. With the number of channels doubled in each convolutional layer, the width and height of the feature maps are halved.



(a) (b)

Figure 5: (a) Training procedure of CDT. (b) Testing procedure of CDT.

In the training procedure of CDT, the input is the cropped defect region, which is appropriately resized to $64 \times 64 \times 3$. We exploit the cross-entropy loss as the loss function $\mathcal{L}_{class}$ in CM:

$$\boldsymbol{softmax}(\boldsymbol{z})_i = \frac{\boldsymbol{e}^{\boldsymbol{z_i}}}{\sum_{\boldsymbol{j=1}}^{\boldsymbol{C}} \boldsymbol{e}^{\boldsymbol{z_j}}}, \tag{8}$$

$$\mathcal{L}_{class} = -\sum_{i=1}^{C} k_i \log\left(softmax(z)_i\right), \tag{9}$$

where $\boldsymbol{softmax}(\boldsymbol{z})$ refer to the probabilities indicating different categories , $\boldsymbol{C}$ is the number of classes (2 in our experiments), and $\boldsymbol{k_i}$ is a ground-truth indicator for class $\boldsymbol{i}$. During the testing, by matching $\boldsymbol{x}$ and $\hat{\boldsymbol{y}}$, the detected region $\boldsymbol{x_{crop}}$ is fed into the well-trained classifier. The ultimate output is the category owning the highest probability:

$$\hat{c} = \arg\min_i softmax(Z), \tag{10}$$

In brief, the test procedure of the three stages for RSDD is presented in Algorithm 1. With the parameters optimized, the test sample $x \in X_{test}$ sequentially pass through the whole pipeline to finally get the segmentation map $\hat{y}$ and the categories $C: \{c^0, c^1, \ldots, c^K\}$ of different defects in each image.

Algorithm 1 Test procedure for RSDD.

Input: Real image $x \in X_{test}$, ground truth $y \in Y_{test}$, three functions with trained parameters $iden()$, $seg()$, and $cla()$.

Output: Segmentation map $\hat{y}$ and the class of defects C.

1. $\hat{x} = iden(x)$
2. $score(x, \hat{x}) = \mathbb{E}_{x,\hat{x}} \left\| f_{E_\theta}(x) - f_{G_\phi}(\hat{x}) \right\|_1$
3. **if** $score > threshold$ **then**
4. $\quad diff(x, \hat{x}) = \|x - \hat{x}\|_1 + \sum \left\| f_{i,E_\theta}(x) - f_{i,G_\phi}(\hat{x}) \right\|_2$
5. $\quad \hat{y} = seg(diff(x, \hat{x}))$
6. $\quad$ crop out the highlighted regions from x as $\{x_{crop}^0, x_{crop}^1, \dots, x_{crop}^K\}$
7. $\quad$ **for** $i = 0,1, \dots, K$ **do**
8. $\quad\quad c^i = cla(x_{crop}^i)$
9. $\quad$ **end for**
10. $\quad$ **return** $\hat{y}$ and $C: \{c^0, c^1, \dots, c^K\}$
11. **end if**

4 Computational Experiments

In this section, the computational experiments of the proposed framework for RSDD are conducted. Firstly, we introduce the dataset used in the experiments. Then, we report the evaluation metrics and training setup in the three stages. Finally, the results of each stage as well as the whole pipeline are collected.

4.1 Rail Surface Dataset

In our experiment, the dataset of rail images is sponsored by the Hong Kong Metro Corporation (MTR) and Beijing Jiaotong University. Since the raw images contain the rails and other components, such as the bolts and clips, we crop them to discard the unrelated areas outside the rails. In dataset D_H, there are 3490 healthy rail images collected by two cameras mounted on the train bottom. D_H is divided into D_H^{train}, D_H^{val}, and D_H^{test} with a ratio of 8:1:1 for the evaluation of the IUR stage. Besides, another dataset D_{UH} containing unhealthy rail images with different surface defects is also collected for the training and validation of SDR and CDT.

Due to the rare occurrence of surface defects, we apply data augmentation technologies to overcome the limitation of the railway dataset itself and ensure the robustness of the three stages. Deformation, scaling, and gray value variations are the main methods for the data augmentation of unhealthy rails. As summarized in Table 1, we have 2000 unhealthy rail images (D_{UH}^{train}) utilized for training the SDR, 395 images for validation (D_{UH}^{val}), and 400 images for testing (D_{UH}^{test}). IUR is evaluated with $D_{UH}^{val} + D_{UH}^{val}$. SDR and CDT are evaluated alone with D_{UH}^{val}. At last, a test set $D_{UH}^{test} + D_{UH}^{test}$ is applied to assess the effectiveness and superiority of the whole pipeline.

Stage	train	validation	test
IUR	2792	744	/
SDR	2000	395	/
CDT	2000	395	/
whole pipeline	**/**	**744**	**749**

Table 1: Image summary of three sets.

In the training and testing process, width and height of the rail images are fixed to 128. In the CDT stage, there are two types of surface defects, type-I and type-II, as shown in Figure 1. The type-I defect usually has a narrow and long shape, while the type-II defect owns a larger area and a more irregular contour than the type-I defect. Additionally, each sample in D_{UH} contains at least one defect. These defects may belong to different types. There are totally 579 type-I defects and 216 type-II defects in D_{UH}^{val}, 604 type-I defects and 225 type-II defects in D_{UH}^{test}.

4.2 Evaluation Metrics

Different metrics are exploited for various tasks in our proposed framework. Firstly, the IUR is evaluated at an image level. A set of common metrics based on $enc(x, \hat{x})$ are exploited including the area under the precision-recall curve (AUPRC) and area under receiver operating characteristic curve (AUROC). The equal error rate (EER) is another metric to evaluate IUR. It is a mathematical way of the trade-off between FP and FN. The optimal EER point is marked on the ROC curves in our experiments.

The second stage SDR applies pixel-wise evaluation metrics, including pixel accuracy (PA) and intersection over union (IoU). However, in the ground truth map of anamalous image, most of the pixels are black, representing background, while the defect region occupies only a small area. In this case, we specially record the average pixel accuracy (APA) to balance the segmentation capability on both defect region and background. In our RSDD task, the IoU of the image is calculated only on the class of defect (DIoU).

During the CDT stage, the confusion matrix of the two types of defects is applied. Accuracy, precision, recall, and F1-score values can also be calculated with the confusion matrix. If not specified, all above metrics, except for EER, are reported as percentages to provide a fair comparison. During testing, the whole pipeline works in precisely the same manner as during the training stage. Besides, the speed of each stage and the entire framework during testing is recorded in terms of average time (ms).

4.3 Training Setup

The first stage IUR is trained with a semi-supervised approach that only healthy rails are fed into the network. It is optimized based on $\mathcal{L}_{ident}$, using AdamW with momentums $\beta 1 = 0.5, \beta 2 = 0.99$, while the initial learning rate is set to $lr = 10^{-3}$. Experiments are conducted with three NVIDIA GeForce RTX 2080 GPUs. The batch size of each training iteration is 18 with each GPU assigned to process 6 images. The whole training schedule is empirically set to 200 epochs to yield optimal results.

We keep the initial learning rate for the first 100 epochs and linearly decay it to zero in the next 100 epochs.

The SDR stage employs supervised training to feed the ground truth labels and rail images into the network. The discriminator G_ϕ and the AE part are locked into fierce competition. They are trained in alternating steps, following the objective in (7). The iteration of D_ω is set to three for each iteration of AE architecture to make sure D_ω not win early in the competition, and the gradient will not vanish. SGD optimizer is used in the training process with an initial learning rate $lr = 2e - 4$. After the parameters of D_ω are updated, their values will be truncated to between -0.01 and 0.01 to prevent the loss value from rising continuously. With the same three NVIDIA GeForce RTX 2080 GPUs, SDR owns a batch size of 24, trained for 200 epochs.

Finally, CDT adopts supervised training as well and utilizes Adam as the optimizer with a learning rate of 10^{-4}. The categorical cross-entropy $\mathcal{L}_{class}$ is exploited as loss function. The whole training schedule is set to 200 epochs. Batch size is fixed to 24 with each GPU assigned to process 8 images.

4.4 Results and Analysis

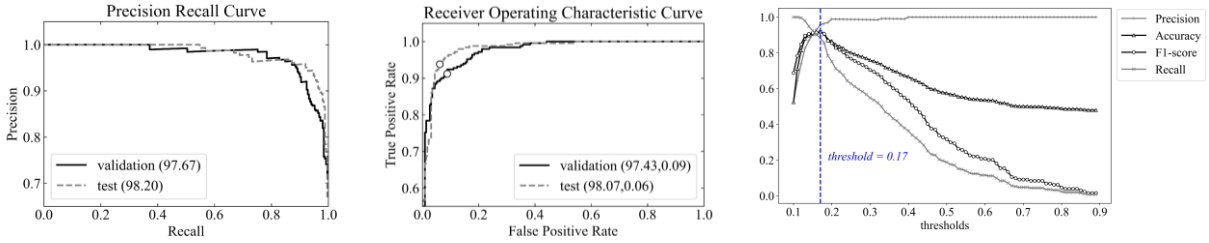
In this section, the experimental evaluation of the proposed framework is conducted. Each stage is validated using the image sets divided in Table 1. First, we compare the identification performance with four different choices of the match function $M \in \{L_1, L_2, bottle(x, \hat{x}), enc(x, \hat{x})\}$. L_1 and L_2 denotes the L1 distance and L2 distance between x and $\hat{x}$, respectively. $bottle(x, \hat{x})$ denotes the bottleneck features [29]. Table 2 summarizes the validation results of different defect scores. $bottle(x, \hat{x})$ and $enc(x, \hat{x})$ show close performance on the validation set while the L1 and L2 residuals have significantly worse performances than other two indicators. The encoded indicator $enc(x, \hat{x})$ shows its superiority in the architecture of IUR.

Score	**AUPRC**	**AUROC**	**EER**	**Time(ms)**
L_1	94.69	93.60	0.15	6.15
L_2	96.18	95.74	0.11	6.15
$bottle(x, \hat{x})$	97.35	96.92	0.09	6.15
$enc(x, \hat{x})$	**97.67**	**97.43**	0.09	6.15

Table 2: Validation performance of four score methods.

Next, the PR curve and ROC curve from the validation and testing process of IUR are displayed in Figure 6(a-b), in which the validation results and test results are close. The AUPRC values are generally higher than AUROC because there are more unhealthy samples than healthy rails in $D_{UH}^{val} + D_H^{val}$ (395 vs. 349) and $D_{UH}^{test} + D_H^{test}$ (400 vs. 349). Figure 6 indicates that the IUR stage is capable of easily identifying the unhealthy rails the healthy ones. As the unhealthy rail is decided according to $enc(x, \hat{x})$, a valid threshold in Algorithm 1 is required to activate the next stage during testing. Figure 6(c) shows the decision process of the score threshold. As the threshold increases, metrics including the F1-score and accuracy rise first and then fall, reaching their maximum values when the threshold equals 0.17. Thus, test rail images with

$enc(x, \hat{x})$ greater than 0.17 will be identified as unhealthy, and the SDR is activated simultaneously.



(a) (b) (c)

Figure 6: Evaluation of IUR. (a) AUPRC. (b) AUROC and EER. (c) Decision process of the threshold based on $enc(x, \hat{x})$.

The design of SDR for developing the segmentation results is discussed in Figure 7. We compare the generic AE backbone and the U-net backbone, which allows low-level information from E_θ to shortcut across the network on the surface defect segmentation. The same discriminator D_ω is cascaded at the end of both U-net and AE to ensure a fair comparison. Figure 7 demonstrates that a simple AE is easy to ignore the small defect areas and detect them as background (black). The advantages of the U-net appear not to be specific to type-II defects that are larger and more irregular. However, when there are type-I defects with small sizes, the U-net again achieves the superior results. These observations are quantified using the accuracies and DIoU as in Table 3. As mentioned in Section 4.2, we focus on not only the PA on the image (which is imbalanced), the APA on both background and defect can reflect the segmentation effect of pixels belonging to defects more intuitively. Table 3 shows that the three metrics vary a lot in both the validation set and test set. It is reduced by almost 20% compared with the PA on image level when calculating the APA. The DIoU reaches a middle value between PA and APA, indicating that the DIoU can be treated as a balanced measurement between the two accuracies. The U-net achieve higher values for both APA and DIoU, revealing that the concatenation between E_θ and G_ϕ in SDR generator enhances producing nearly the accurate segmentation regardless of input defect type.

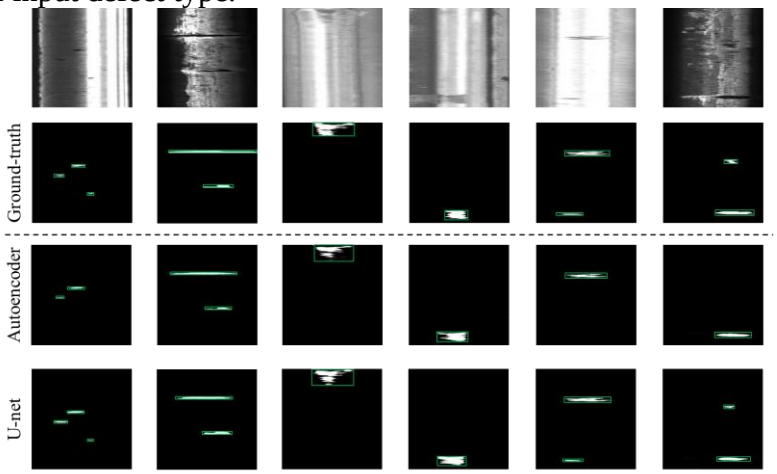


Figure 7: Segmentation performance of AE and U-Net.

Backbone	Image sets	**PA**	**APA**	**DIoU**	**Time(ms)**
Autoencoder	$\boldsymbol{D_{UH}^{val}}$	91.44	73.86	80.26	6.94
	$\boldsymbol{D_{UH}^{test} + D_{H}^{test}}$	90.28	69.01	75.94	
U-net	$\boldsymbol{D_{anom}^{val}}$	**97.39**	**79.53**	**85.68**	10.36
	$\boldsymbol{D_{UH}^{test} + D_{H}^{test}}$	96.76	74.82	80.37	

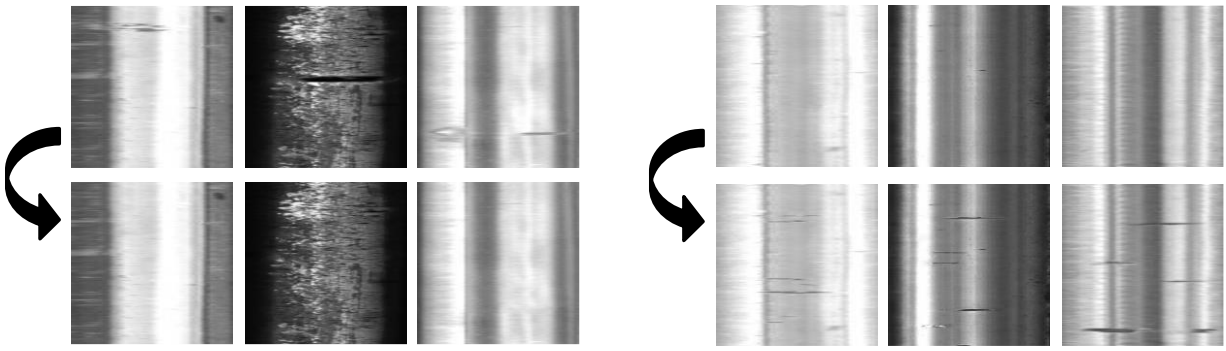
Table 3: Segmentation results on different image sets.

Table 4 summarizes the classification results when CDT is validated alone with D_{UH}^{val} as well as when tested in the whole pipeline with $D_{UH}^{test} + D_{H}^{test}$. Generally, the precision of type-II defects is much smaller than the type-I value indicating that our classifier is easier to misclassify the type-I defects as type-II. In testing, the number of overall classifications is slightly less than the number of defects in D_{H}^{test} because of the influence of the previous stages. Nevertheless, the testing performance is still comparable to the validation results.

Actual	Predicted		**Precision**	**Recall**	**F1-score**	**Time(ms)**
	type-I	type-II				
	$\boldsymbol{D_{UH}^{val}}$					
type-I	566	13	99.30	97.75	98.52	1.02
type-II	4	212	94.22	98.15	96.14	
Accuracy	/		/	/	97.86	
Average			96.76	97.95	**97.35**	
	$\boldsymbol{D_{UH}^{test} + D_{H}^{test}}$					
type-I	588	10	99.16	97.35	98.25	1.28
type-II	5	213	95.52	94.67	95.09	
Accuracy	/		/	/	96.62	
Average			97.34	96.01	**96.67**	

Table 4: Classification results on different image sets.

Finally, to validate the applicability of the proposed framework, comparative experiments are conducted with a new dataset D_{syn}. Since there are few public rail datasets about the defect inspection, we synthesize the unhealthy rails in D_{syn} by copy-pasting [30] the defect regions of type-I and type-II to the D_H as well as synthesize the healthy rails in D_{syn} by removing the defects area in D_{UH}. Moreover, the number of defects in D_{syn} is increased with more than one paste. Figure 8 shows some examples of the synthesized images in D_{syn}. We divide the D_{syn} into D_{syn}^{val} containing 205 unhealthy rails with totally 630 defects and D_{syn}^{test} containing 231 unhealthy rails with totally 674 defects. Table 5 displays the validation results and test results of the proposed framework on two datasets. With more defects, the unhealthy rails are more accessible to recognize according to the performance of IUR stage. The new dataset also owns better segmentation and classification performance with more frequent defects. Overall, the framework is able to run all stages at the same time at 19.30ms. Tests have been carried on a PC with an Intel i7-8700 and an NVIDIA GeForce RTX 2080 GPU. Results in Table 5 demonstrate that the proposed framework can deal with various targets in the RSDD scenarios.



(a) (b)

Figure 8: (a) Healthy rails in D_{syn} and (b) Unhealthy samples in D_{syn}.

Image sets	Identification		Segmentation		Classification		Time(ms)
	AUPRC	**AUROC**	**APA**	**DIoU**	**type-I**	**type-II**	
$\boldsymbol{D_{UH}^{val} + D_{H}^{val}}$	97.67	97.43	72.60	79.21	96.84	94.90	19.30
$\boldsymbol{D_{UH}^{test} + D_{H}^{test}}$	98.20	98.07	74.82	80.37	98.25	95.09	
$\boldsymbol{D_{syn}^{val}}$	99.17	99.05	79.31	87.00	98.04	96.83	20.07
$\boldsymbol{D_{syn}^{test}}$	99.86	99.60	80.77	88.45	98.95	98.02	

Table 5: Performance of the whole pipeline based on two datasets.

5 Conclusions

We presented a unified framework in this paper for addressing multiple analytical functions in RSDD. The framework was composed of three sequential stages, such as the IUR, SDR, and CDT. Each of the stages was developed with deep learning architecture. The effectiveness of each stage was validated through computational experiments. We also synthesized a new dataset to confirm the applicability of the proposed framework. The performance on the new dataset demonstrated that the framework was capable of dealing with various targets in more complex RSDD scenarios.

References

[1] R. Edwards, S. Dixon, and X. Jian, "Characterisation of defects in the railhead using ultrasonic surface waves," *NDT & e International*, vol. 39, no. 6, pp. 468-475, 2006.

[2] V. Sulimova, A. Zhukov, O. Krasotkina, V. Mottl, and A. Markov, "Automatic Rail Flaw Localization and Recognition by Featureless Ultrasound Signal Analysis," in *International Conference on Machine Learning and Data Mining in Pattern Recognition*, 2018: Springer, pp. 16-27.

[3] X. Ni, H. Liu, Z. Ma, C. Wang, and J. Liu, "Detection for Rail Surface Defects via Partitioned Edge Feature," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[4] 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)*, 2015: IEEE, pp. 1-6.

[5] S. Liu, Q. Wang, and Y. Luo, "A review of applications of visual inspection technology based on image processing in the railway industry," *Transportation Safety and Environment*, vol. 1, no. 3, pp. 185-204, 2019.

[6] H. Di, X. Ke, Z. Peng, and Z. Dongdong, "Surface defect classification of steels with a new semi-supervised learning method," *Optics and Lasers in Engineering*, vol. 117, pp. 40-48, 2019.

[7] X. Zhang, N. Feng, Y. Wang, and Y. Shen, "Acoustic emission detection of rail defect based on wavelet transform and Shannon entropy," *Journal of Sound and Vibration*, vol. 339, pp. 419-432, 2015.

[8] E. T. Esfahani, S. Wang, and V. Sundararajan, "Multisensor wireless system for eccentricity and bearing fault detection in induction motors," *IEEE/ASME Transactions on Mechatronics*, vol. 19, no. 3, pp. 818-826, 2013.

[9] R. Gasparini, A. D'Eusanio, G. Borghi, S. Pini, S. Giuseppe, S. Calderara, F. Eugenio, and R. Cucchiara, "Anomaly Detection, Localization and Classification for Railway Inspection," in *25th International Conference of Pattern Recognition*, 2020.

[10] A. K. Singh, A. Swarup, A. Agarwal, and D. Singh, "Vision based rail track extraction and monitoring through drone imagery," *ICT Express*, vol. 5, no. 4, pp. 250-255, 2019.

[11] C. Mandriota, E. Stella, M. Nitti, N. Ancona, and A. Distante, "Rail corrugation detection by Gabor filtering," in *Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205)*, 2001, vol. 2: IEEE, pp. 626-628.

[12] 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.

[13] Q. Li and S. Ren, "A real-time visual inspection system for discrete surface defects of rail heads," *IEEE Transactions on Instrumentation and Measurement*, vol. 61, no. 8, pp. 2189-2199, 2012.

[14] 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.

[15] K. Ma, T. F. Y. Vicente, D. Samaras, M. Petrucci, and D. L. Magnus, "Texture classification for rail surface condition evaluation," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016: IEEE, pp. 1-9.

[16] S. Ghorai, A. Mukherjee, M. Gangadaran, and P. K. Dutta, "Automatic defect detection on hot-rolled flat steel products," *IEEE Transactions on Instrumentation and Measurement*, vol. 62, no. 3, pp. 612-621, 2012.

[17] 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)*, 2016: IEEE, pp. 2584-2589.

[18] L. Song, W. Lin, Y.-G. Yang, X. Zhu, Q. Guo, and J. Xi, "Weak micro-scratch detection based on deep convolutional neural network," *IEEE Access*, vol. 7, pp. 27547-27554, 2019.
[19] T. Wang, F. Yang, and K.-L. Tsui, "Real-Time Detection of Railway Track Component via One-Stage Deep Learning Networks," *Sensors*, vol. 20, no. 15, p. 4325, 2020.
[20] Y. Zhang, M. Liu, Y. Yang, Y. Guo, and H. Zhang, "A Unified Light Framework for Real-time Fault Detection of Freight Train Images," *IEEE Transactions on Industrial Informatics*, 2021.
[21] D. Zhang, K. Song, Q. Wang, Y. He, X. Wen, and Y. Yan, "Two Deep Learning Networks for Rail Surface Defect Inspection of Limited Samples with Line-Level Label," *IEEE Transactions on Industrial Informatics*, 2020.
[22] 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.
[23] T. Wang, Z. Zhang, F. Yang, and K.-L. Tsui, "Intelligent Railway Foreign Object Detection: A Semi-supervised Convolutional Autoencoder Based Method," *arXiv preprint arXiv:2108.02421*, 2021.
[24] 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*, 2015: Springer, pp. 234-241.
[25] C. Lin, S. Zhang, S. You, X. Liu, and Z. Zhu, "Real-time foreground object segmentation networks using long and short skip connections," *Information Sciences*, vol. 571, pp. 543-559, 2021.
[26] M. Shao, G. Zhang, W. Zuo, and D. Meng, "Target attack on biomedical image segmentation model based on multi-scale gradients," *Information Sciences*, vol. 554, pp. 33-46, 2021.
[27] 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*, 2017, pp. 1125-1134.
[28] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*, 2017: PMLR, pp. 214-223.
[29] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "Ganomaly: Semi-supervised anomaly detection via adversarial training," in *Asian conference on computer vision*, 2018: Springer, pp. 622-637.
[30] M. Kisantal, Z. Wojna, J. Murawski, J. Naruniec, and K. Cho, "Augmentation for small object detection," *arXiv preprint arXiv:1902.07296*, 2019.