Applied Soft Computing 111 (2021) 107706

Contents lists available at ScienceDirect

Applied Soft Computing

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

# Defect classification based on deep features for railway tracks in sustainable transportation

Ilhan Aydin [*], Erhan Akin, Mehmet Karakose

*Computer Engineering Department, Firat University, 23119, Elazig, Turkey*

ARTICLE INFO

*Article history:*
Received 12 December 2020
Received in revised form 30 June 2021
Accepted 5 July 2021
Available online 14 July 2021

*Keywords:*
Railway surface defect
Visual inspection
Deep learning
Image processing
Railway maintenance

ABSTRACT

Rail tracks are the most important component of train movement in rail transportation. Therefore, real-time detection of defects on track surfaces is important but also difficult because of the noise, low contrast, and inhomogeneity of density. In recent years, tools have been developed for robust and highly accurate defect detection with advances in deep learning technologies. However, the existing deep learning algorithms require a large number of parameters to be set, which is computationally expensive. Therefore, those algorithms cannot fulfill the requirements for quick inspection. In this study, rail surface defects were detected by fusing the features of two deep learning models. SqueezeNet and MobileNetV2, the two models selected for this purpose, are both smaller in size and faster than other deep learning models. However, both of these models are less accurate than other models. Therefore, in this study, a fusion model with high accuracy is proposed by combining the features of the two models. First, a contrast adjustment is applied to the original image of the rail, and then the rail track location is determined. Then, most weighted features are selected from each network, and the defects are determined by giving the reduced features to Support Vector Machines (SVM). Experimental results show that the proposed method gives better results for multiple rail surface defects under low contrast than using a single deep learning model.

## 1. Introduction

Worldwide, railway transportation has expanded in recent years and has become the most preferred means of transportation. Safe train travel depends on the quality of the railway tracks. Therefore, detection of railway defects is important and should be detected at an early stage. Contact measurement techniques and human inspection are still used to find rail failures [1]. A trained engineer periodically checks the rail components by walking on the rail line. The main disadvantages of manual inspection are early detection, reliability, and the cost of detection systems. In addition, these techniques have low sensitivity and accuracy rates and do not meet the needs of modern railway technology. With the widespread use of railways, more serious controls for transportation safety are required [2]. Therefore, techniques to automatically inspect the rails, shorten the control time and reduce maintenance costs are needed to increase railway transportation safety [3]. In recent years, improvements in camera resolution and frame rates as well as improvements in image processing, computer vision, and machine learning methods have made important contributions to automating railway maintenance.

Many researchers have studied automatic inspection systems and methods to facilitate railways maintenance. For this purpose, nondestructive methods such as acoustic emission [4], eddy current [5], and ultrasonic perception [6] have been proposed to detect railway defects. Eddy current was used to locate rail defects [7]; however, it is very difficult to detect rail failures at different levels with this method. An acoustic emission-based method has been proposed to detect rail faults, and a multi-level adaptive noise cancellation method has been presented to eliminate the generated noise [8]. To increase the effectiveness of the proposed method in this paper, the features of noise and defect signals were analyzed. The ultrasonic perception technique has been proposed to detect cracks on the rails. For this purpose, ultrasonic waves that occurred during motion were analyzed, and contactless laser probing was used to detect flaws [9]. Although these non-contact techniques are used to identify rail faults, they are difficult to use in practical terms. Computer vision-based algorithms have the ability to detect the rail components and determine the faults from the images taken, simultaneously. In computer vision-based inspection systems, a high-resolution camera is placed under the measuring train, and the rail images

* Corresponding author.
*E-mail addresses:* iaydin@firat.edu.tr (I. Aydin), eakin@firat.edu.tr (E. Akin), mkarakose@firat.edu.tr (M. Karakose).

https://doi.org/10.1016/j.asoc.2021.107706
1568-4946/© 2021 Elsevier B.V. All rights reserved.

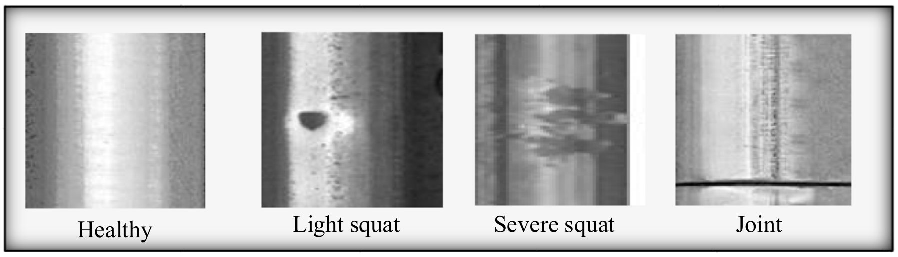


**Fig. 1.** The types of railway surface defects.

are acquired as the train moves. The collected images are analyzed on-site in real time or later by image processing methods on a computer. Problems related to rail components are classified as rail cracks, corrugations on the rails, bolt defects, fastener defects, and rail surface defects [7–11]. Rail surface defects occur due to faults during production, misuse, and rolling contact fatigue. These defects are difficult to detect because of their complex formation mechanisms and their effects on the rail. Therefore, it is one of the most important defects that require early intervention. Rail surface defects are divided into two classes as: corrugation and discrete. Corrugation is the wear that occurs on the rail in a certain period. Discrete defects are randomly formed defects that do not meet certain characteristics. The most common rail failures can be classified as cracks, squats, and connection problems [3]. Fig. 1 shows different types of railway surface defects.

In Fig. 1, one of the defects is the squat and rail defect, which occurs in the running band on the rails. Another type of defect is the insulation joint problem, which breaks the connection between two rails. The change of defects on the rail surface is not linear. The defects that occur get bigger as the railway is used. In addition, existing defects can be used to predict future defects. Therefore, the inspection vehicle must be run periodically on railways [10]. Automated vision-based rail defect detection methods are summarized in Table 1.

In Table 1, preprocessing in the methods proposed in the literature for determining rail defects, the detection method used, and the advantages and disadvantages of the methods are analyzed. In addition, it is also given in the methods whether the region of interest (ROI) of the rail is extracted. All of the methods using preprocessing are aimed at removing the noise and illumination effects by applying them directly to the rail surface. In addition, many studies have not been able to obtain the rail surface from the image. The detected rail defects were considered as a single class, and the defective area was determined based on segmentation [20–22]. Many of these methods have low sensitivity to noise, and their defect detection performance is not sufficiently accurate [23,24]. Most recent studies have detected defects on the rail surface using the features of gray images. However, the rail surface is noisy and does not have a uniform distribution in the images due to the lighting used for image acquisition. Therefore, detecting rail surface defects is still seen as a challenge. Detection of rail surface defects is a challenging problem with a visual inspection system for the following reasons.

- *Irregular illumination on the rail surface*: The image collection system used in measurement trains collects images with cameras placed under a wagon. Although a lighting system is used with measurement systems, the brightness and contrast of the rail surface images will change in natural light and weather conditions [3]. In addition, the measuring system is affected by the vibrations generated by the moving train.
- *Changes in reflections on the rail surface*: Although the light reflection feature of the rail surface is good, the reflections are not the same on all sides because the rail head has a non-flat convex structure. The middle part of the rail appears brighter due to the reflection, while the sides appear darker [1]. This makes it difficult to distinguish defects and backgrounds.
- *Random distribution of rail surface defects*: Different defects occur on the rail surface, and these defects show a different distribution. Therefore, many vision-based methods use gradient information of defective regions by analyzing the gray-scale rail image [18]. However, misleading surfaces, such as rust and stains, cause vision-based methods to misinterpret the images.
- *Difficulties in generating data sets for different defect types*: In practice, collecting different defect types takes a long time. Therefore, most of the studies in the literature determine if there are defects on the rail surface rather than determining the types of surface defects. For example, a joint problem is a serious fault and requires early intervention.

To address the problems mentioned above, this paper introduces a new image processing and deep learning-based approach to detect and classify rail defects. The proposed approach basically consists of three steps. In the first step, a new image enhancement method is introduced to ensure the consistency of the collected images and to reduce the effects of noise. The second phase determines the rails in the image to avoid unnecessary calculations during the final defect classification stage. Based on the received region of interest (ROI), decision making is presented by combining the features of two different deep learning models. The main contribution of the paper is to fill the following gaps in railway surface defect detection: (i) enhancing the image to provide more reliable rail track extraction and not requiring any parameter adjustments; (ii) extracting appropriate features and reducing the dimension to increase the performance of deep learning-based defect classification models; (iii) training deep learning classifiers to reduce human involvement in defect detection; and (iv) comparing the proposed method with single deep learning models and obtaining better results.

The rest of the paper is organized as follows. In Section 2, the image enhancement and preprocessing steps are investigated, and a new method is proposed to eliminate the illumination problems. Also, a rail positioning algorithm is given in Section 2. Section 3 introduces the feature extraction from two combined Convolutional Neural Networks (CNN) and defect classification based on Support Vector Machines (SVM). Experiments and defect inspection results are given in Section 4. Also, comparative analyses are discussed in this section. Finally, the conclusions are presented in Section 5.

## 2. Image preprocessing and rail positioning

There are ballasts, fasteners, sleepers, and other parts expected in the images of the rail region collected by the camera in the

**Table 1**
Evaluation of automated visual inspection methods for defect detection of rail surfaces.

| Ref. | Used Preprocessing | Defect detection method | Advantages/Disadvantages | ROI |
|---|---|---|---|---|
| [1] | Gray contrast equalization | Curvature filter and Gaussian mixture model based segmentation | – Advantages: Eliminating the noise and robust background modeling and high accuracy<br>– Disadvantages: complex computing and low frame rate | Yes |
| [12] | Local weber like contrast enhancement | Wavelet analysis based gray stretch maximum entropy | – Advantages: eliminating the effects of non-uniform illumination and good segmentation results<br>– Disadvantages: In complex background, the efficiency of the model decreases | Yes |
| [13] | Not any preprocessing | Transfer learning based on Yolov3 and RetinaNet | - Advantages: Good recall and precision performance on the limited data set<br>- Disadvantages: The method requires different levels of crack size thresholds. | No |
| [14] | Local Michelson-like contrast enhancement | Emphasized maximum entropy thresholding based segmentation | – Advantages: A simple image processing methods based defect detection<br>– Disadvantages: A threshold value should be defined for eliminating the noises. | No |
| [15] | Gamma correction based image enhancement | Background and saliency based defect detection for different scales | – Advantages: Line-by-line investigation of background and defective parts and sensitivity to noise<br>– Disadvantages: Low contrast defects may not be detected when the ray has multiple defects | No |
| [16] | Log operator and z-score method | Pixel level based background modeling | – Advantages: More accurate background modeling by using randomly selected pixels<br>– Disadvantages: All pixels should be used to construct background model | No |
| [17] | Not any preprocessing | Morphological pyramid for defect detection | – Advantages: The model is insensitive to the change of brightness<br>Disadvantages: The performance criteria achieved are still low and need to be improved. | No |
| [18] | Edge detection and thresholding edges | Edge growing and contour filling | – Advantages: The proposed partitioned edge features eliminates the illumination condition and building a homogeneous background<br>– Disadvantages: The proposed method uses a threshold value to obtain defected region and different kinds of defect still cannot be done. | Yes |
| [19] | Not any preprocessing | Gaussian mixture models based segmentation and Faster RCNN based defect detection | – Advantages: Noise sensitive segmentation, automatic defect detection, good performance on different data set<br>– Disadvantages: Real-time implementation difficulty due to combining three complex models | No |

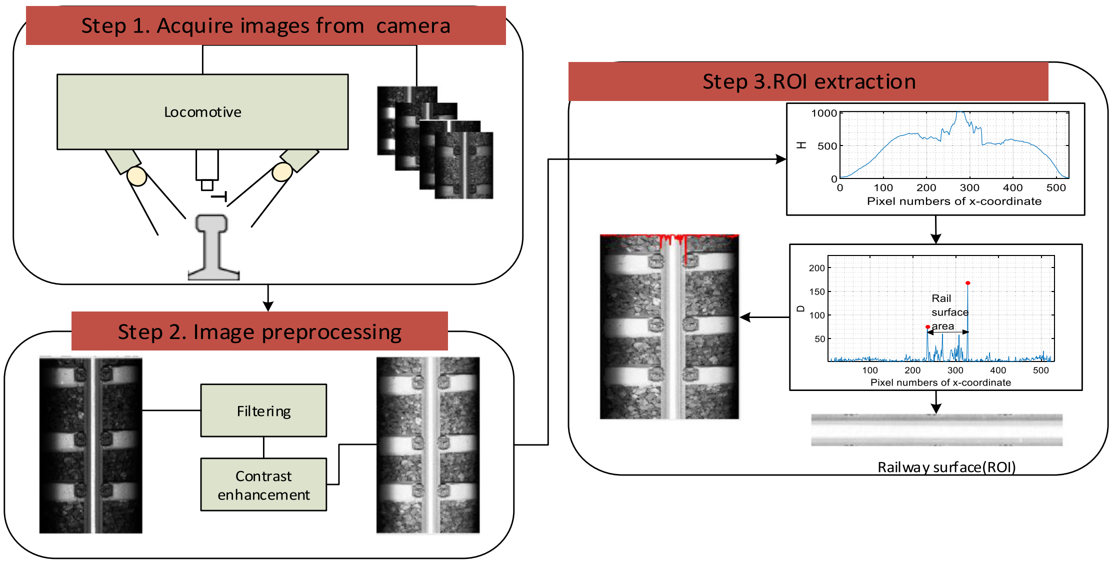


**Fig. 2.** Image preprocessing and ROI extraction.

image collection system. In order to detect rail surface defects, the position of the rail in the image must be found and cropped. However, images taken under the train are often subject to noise and lighting problems. For this reason, preprocessing should first be done to the obtained images to eliminate the problems related to lighting. Then, the rail surface will be determined by the rail positioning algorithm. A block diagram of the proposed preprocessing and rail positioning method is given in Fig. 2.

In Fig. 2, two vertically placed cameras capture images of the rail surfaces in the first stage. These cameras are mounted below a locomotive and each camera monitors a side of the rail. For each rail, two light sources are installed to reduce the effect of variable lighting conditions. In the second stage, image processing methods such as filter and contrast enhancement are applied to the images obtained in order for the rail detection algorithms to work correctly. The ROI-containing rail is extracted from the obtained image in the third step. The ROI images are saved in a database and used for the defect recognition module. The details of image preprocessing and ROI extraction are given in the following subsections.

### *2.1. Image enhancement and preprocessing*

A color image is kept in three matrices, called R, G, and B. These matrices hold the intensity values of the Red (R), Green (G), and Blue (B) components of an image. The intensity value of an (x, y) pixel in any image I is expressed in R (x, y), G (x, y), and B (x, y), respectively. In other words, a pixel consists of three density values. The width and height of an image is shown by $M$ and $N$ for $x = 1, \ldots, M$ and $y = 1, \ldots, N$, respectively. Although there are many methods for converting an RGB image to a grayscale format, the generally used method is given in (1) [25].

$$I(x,y) = c_r R(x,y) + c_g G(x,y) + c_b B(x,y) \tag{1}$$

In (1), $c_r$, $c_g$ and $c_b$ values are the coefficient values of each component of the RGB color image. The I $(x, y)$ represents the intensity value of the gray image at the point $(x, y)$. When a colored RGB image is converted to a gray format, coefficient values cr, cg, and cb are taken as 0.2989, 0.5870, and 0.1140, respectively [26]. Hue and saturation information is lost while brightness is preserved after this conversion [27]. First, with image normalization, the image with pixel values between 0 and 255 in gray format is converted to a double format between 0 and 1 as shown in (2).

$$J(x,y) = \frac{I(x,y) - I_{\min}}{I_{\max} - I_{\min}} \tag{2}$$

In (2), the parameters $I_{max}$ and $I_{min}$ show the minimum and maximum intensity value in the image I, respectively. Normalization shrinks the values of the J image between 0 and 1. An appropriate representation of the image is measured with high contrast and low homogeneity. Therefore, an image with high contrast and low homogeneity should have a high variance. For this purpose, the most suitable coefficients should be determined to maximize the variance of the image J. The value of $c_b$ can be taken as 1 without any loss of generalization. For transformation, the $c_r$ and $c_g$ values should be determined to maximize the variance of the image J, and the gradient descent method can be used for this purpose. This method is based on the iterative addition of a small value to the parameters, starting with an initial value, until the most suitable solution is found. The gradient descent is given in (3).

$$(c_r^{i+1}, c_g^{i+1}) = (c_r^i, c_g^i) + (\Delta_r^i, \Delta_g^i) \tag{3}$$

In (3), $\Delta_r^i$ and $\Delta_g^i$ are calculated using the gradient of the variance, respectively. The iteration is stopped when a certain optimum result is achieved. For this purpose, the *fminsearch* function used in MATLAB can be used by taking the negative of the variance in order to maximize the cost function [28]. This function is based on a multidimensional unconstrained optimization technique using the Nelder–Mead method and is very fast. The general structure of the contrast enhancement method is given in Fig. 3.

```
Algorithm: J= function contrast_enhancement(I)
    Input: I→An RGB Image
    Output: J→High contrast gray image
        (c_r, c_g)=arg max_{c_r, c_g} (σ_J^2)
        R=I(:, :,1) %Red component of I
        G=I(:,:,2) % Green component of I
        B=I(:,:,3) % Blue component of I
        if size(I, 3)==1 then
            In=I;
        else
            In=imresize(In,[64 64]);
            c=fminsearch('StdMonochrome', [1 1], [ ],In);
            c_r=c(1)
            c_g=c(2)
            I=c_r*R+c_g*G+B
        endif
        J=(I-min(I(:)))/max(J(:));
    endfunction
```

**Fig. 3.** The algorithm of contrast enhancement.

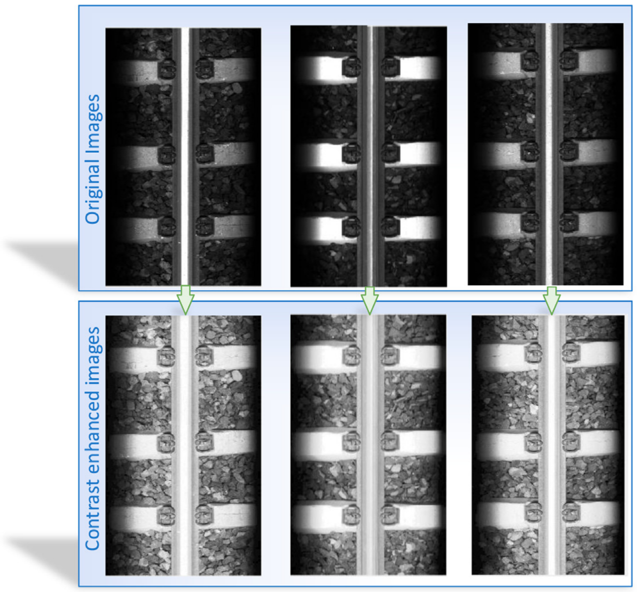


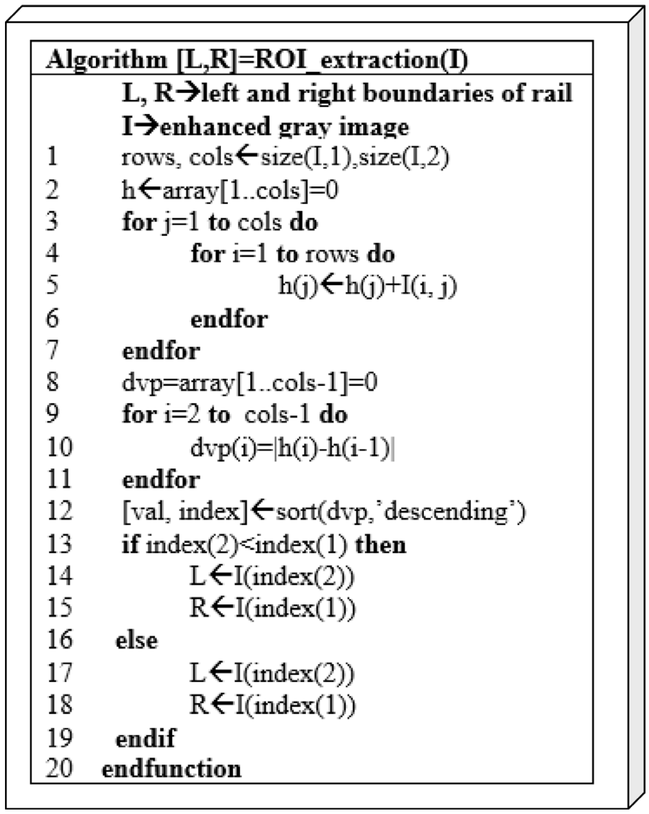
**Fig. 4.** Contrast enhancement for three rail images.

In Fig. 3, an image with high contrast is obtained, and the rail position can be determined more easily in this image than the original image. With the proposed algorithm, the RGB color image taken as a parameter is converted into a gray image, and the optimum coefficients of the red and green channels are calculated to obtain a high contrast image. Fig. 4 shows the results of the contrast enhancement algorithm for different rail images.

As shown in Fig. 4, although the original rail images are taken by placing a light source under the measurement train, very bright regions are still formed on the rail surface, and there is no uniform distribution of the light. In this way, it is difficult to detect the rail. But when contrast enhancement is applied to the original image, it is seen that the rail surface can be clearly distinguished.

### *2.2. Rail positioning*

After the image contrast is enhanced, the ROI must be extracted to identify the rail surface defects. In practice, although



```
Algorithm [L,R]=ROI_extraction(I)
        L, R→left and right boundaries of rail
        I→enhanced gray image
1       rows, cols←size(I,1),size(I,2)
2       h←array[1..cols]=0
3       for j=1 to cols do
4               for i=1 to rows do
5                       h(j)←h(j)+I(i, j)
6               endfor
7       endfor
8       dvp=array[1..cols-1]=0
9       for i=2 to cols-1 do
10              dvp(i)=|h(i)-h(i-1)|
11      endfor
12      [val, index]←sort(dvp,'descending')
13      if index(2)<index(1) then
14              L←I(index(2))
15              R←I(index(1))
16      else
17              L←I(index(2))
18              R←I(index(1))
19      endif
20  endfunction
```

**Fig. 5.** ROI extraction algorithm.

a light source is used when taking the rail images with the measuring train, some parts of the rail are brighter while some parts remain darker due to the placement of the light source. Therefore, this problem is solved with image contrast adjustment. After this step, a method is proposed for cropping the rail surface from the image and extracting the region of interest. The proposed method obtains a vector by making column sums in each row on a gray scale rail image and then finds the two largest values in the signal by taking the absolute difference of the elements of this vector. The obtained values represent the boundaries of the rail. The pseudo code of the ROI extraction is given in Fig. 5.

In Fig. 5, the density of the high resolution image in each line is cumulated in a vector. Then, a difference vector is obtained by taking the difference between consecutive elements of a cumulated vector. The two values where the difference is maximum indicate the rail borders. Fig. 6 demonstrates the application result of the ROI extraction method on a rail image.

As shown in Fig. 6, a signal is obtained to detect the border of the rail track. By taking the difference of successive elements of this signal, the highest two points are taken as the beginning and the end of the rail. Finally, the obtained ROI is given to the image recognition module and the defects are classified.

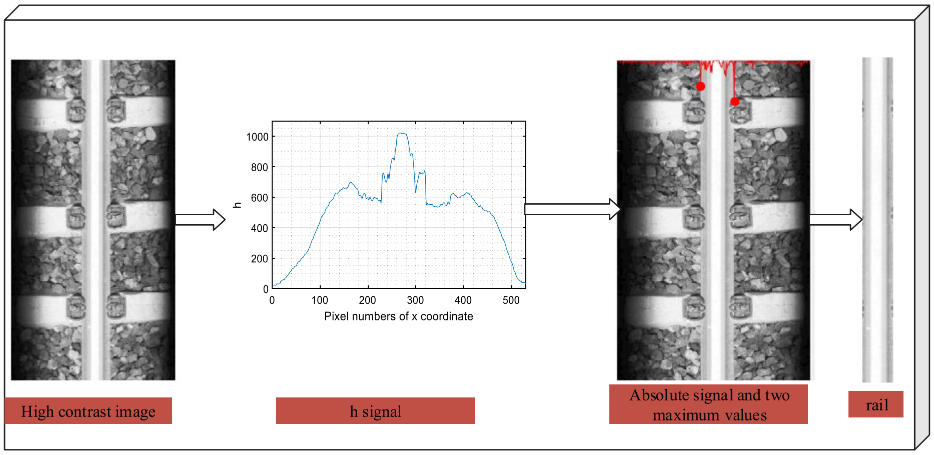
## 3. Classification of rail surface defects using deep features

In the previous section, the problem of image enhancement and rail positioning has been solved, and the resulting image data set consists of only rail surfaces without other rail components. In this section, the deep learning-based feature extraction and classification of defect types will be presented. The defect classification module is created by combining two deep learning modules to achieve a high accuracy rate. The MobileNetV2 [29] and SquezeNet [30] modules were used for this aim. Normally, training deep learning models with transfer learning takes quite a long time and requires many calculations during the testing phase. After the feature extraction with the proposed method, less feature usage will be provided with feature selection via ReliefF [31]. The reason for choosing these two deep learning models is that they require fewer parameters than other deep learning models, and they are faster. MobileNet is a CNN that requires low cost hardware used for classification, segmentation and object detection. The depth-wise convolutional layer proposed in MobileNetV1 has reduced both model size and complexity. In MobileNetV2, on the other hand, the proposed inverted residual blocks removed the non-linear relationship between layers. The MobileNetV2model was developed from MobileNetV1 and has modules that enable linearization between layers as compared to the previous model. SqueezeNet is a CNN consisting of convolution, pooling, ReLU, and Fire layers. SqueezeNet does not have a fully connected layer, and this task is done with Fire layers. SqueezeNet reduces computation time by using pointwise filters and tries to maintain network accuracy by reducing the number of input channels to 3 × 3 filters. It ensures that the convolution layers have large activation maps by sampling late in the network, which results in higher classification accuracy. It has been proposed to combine the two network models, since these two networks both have fewer parameters and can be implemented in mobile or low-cost hardware. Thus, a network model can be created in which defects can be detected in real time. The block scheme of the proposed defect classification method is given in Fig. 7.

In Fig. 7, the features are obtained by giving the training set to each deep neural network. Then, the ReliefF algorithm is applied to determine the weights of the obtained features. A section of the high weight features is selected. These features are then concatenated and given to the support vector machine in order to classify the defects. In recent years, the main focus of the deep learning methods has been accuracy. It is possible to reach a given accuracy rate with multiple CNNs. However, with equivalent accuracy, smaller network architectures require less bandwidth to be implemented embedded, less communication during distributed computing, and can transfer a model from the cloud to autonomous vehicle. SqueezeNet was developed as a CNN architecture that provides an accuracy performance close to Alexnet and has 50 times less parameters. In addition, SqueezeNet uses 1 × 1 filters instead of 3 × 3 filters to reduce the number of parameters, with nine times fewer parameters obtained compared to using 3 × 3 filters. The structure of SqueezeNet deep learning architecture is given in Fig. 8.

The SqueezeNet architecture given in Fig. 8 consists of 16 layers. In the first layer, the input image is given to a convolution layer, and the image passed through the pooling layer is given to eight Fire layers. After the last convolution layer in the network architecture, the network terminates with the ReLU and Softmax layer. In this study, the feature extraction was made from SqueezeNet, and these features were extracted from last trainable layer, called pool10. In this layer, 1,000 features were obtained. The parameters of SqueezeNet are given in Table 2.

The other model to be used for feature extraction is selected as MobileNetV2. This model is especially used in embedded systems due to its low memory usage. MobileNetV2 is a CNN model and consists of 53 layers in total. The model architecture initially starts with a fully convolutional layer with 32 filters, followed by 19 residual bottleneck layers. Here, the inverted residual blocks consist of pointwise and depthwise convolution layers. The pointwise convolution layer is a special form of standard convolution, and the kernel size is one-dimensional for linearly combining different input channels. The inverted residual layer used in MobileNetV2 consists of three separate convolutions. The first is to



High contrast image
h signal
h
Pixel numbers of x coordinate
Absolute signal and two maximum values
rail

**Fig. 6.** Region of interest extraction of rail.

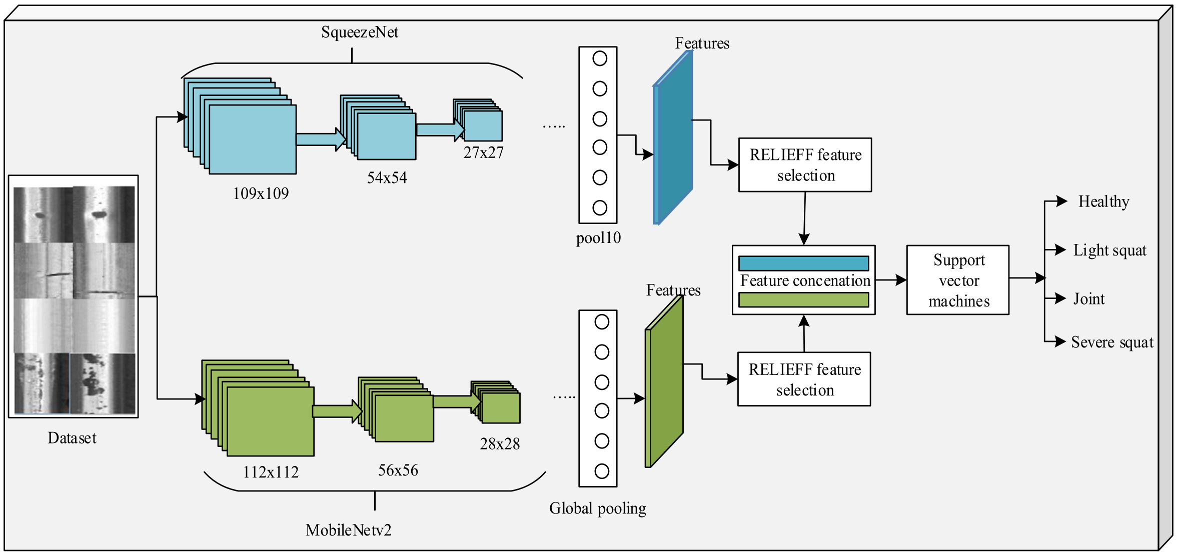


**Fig. 7.** The proposed defect classification method.

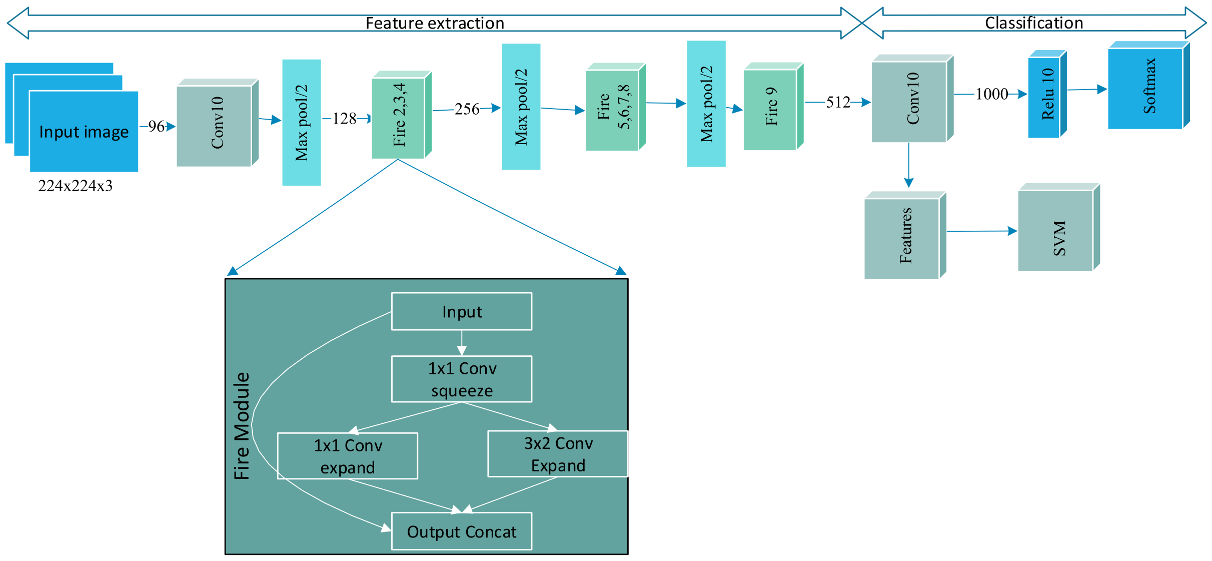


**Fig. 8.** SqueezeNet architecture.

**Table 2**
The SqueezeNet parameters and architectural dimensions.

| Layer name | Output size | Filter size | | |
|---|---|---|---|---|
| | | 1 × 1 squeze | 1 × 1 expand | 3 × 3 expand |
| Input image | 224 × 224 × 3 | – | – | – |
| Conv1 | 111 × 111 × 96 | – | – | – |
| Maxpool1 | 55 × 55 × 96 | – | – | – |
| Fire2, Fire3 | 55 × 55 × 128 | 16 | 64 | 64 |
| Fire4 | 55 × 55 × 256 | 32 | 128 | 128 |
| Maxpool4 | 27 × 27 × 256 | – | – | – |
| Fire5 | 27 × 27 × 256 | 32 | 128 | 128 |
| Fire6, Fire7 | 27 × 27 × 384 | 48 | 192 | 192 |
| Fire8 | 27 × 27 × 512 | 64 | 256 | 256 |
| Maxpool8 | 13 × 12 × 512 | – | – | – |
| Fire9 | 13 × 13 × 512 | 64 | 256 | 256 |
| Conv10 | 13 × 13 × 1000 | – | – | – |
| Avgpool10 | 1 × 1 × 1000 | – | – | – |

use a 1 × 1 pointwise convolution to extend the low-dimensional input feature into a high-dimensional space. After this process, ReLU6 is applied. Then, the depthwise convolution is applied using 3 × 3 kernels, and ReLU6 is applied again. In the last step, the spatial filtered feature map is transformed into a low-dimensional subspace using another pointwise convolution. The structure of MobileNetV2 is given in Fig. 9.

As shown in Fig. 9, the input image passes through 16 block-wise residual blocks after the first convolution block. Afterwards, the input block passes through the ReLU and average pooling block and then reaches the fully connected layer. While the part up to the average pooling block is used for feature extraction, the part after this stage is used for classification. In this study, 1,280 features were obtained from the global average layer and converted to a suitable form for SVM. Table 3 shows the number of filters and other features in the layers.

Some blocks are applied more than once in Table 3. This situation is given in the last column of the table. After the features are obtained from both CNNs, the feature selection is made on these features and the new features are then concatenated to obtain the inputs of the classifier. The mx1000 and mx1280 feature matrices are obtained from the pool10 layer of SqueezeNet and the averagepool layer of MobileNetV2, respectively. Parameter m in the matrix size shows the number of samples in the data set, while the other parameter shows the number of features obtained for each sample. Assuming that the $n_1$ and $n_2$ features are obtained from CNNs after feature selection, the size of the data set to be given to the classifier at the end of the concatenation is mx $(n_1 + n_2)$. The ReliefF method is used for feature selection. This method generates negative and positive weights for the features by selecting the distance-based selection, and then the negative weights are eliminated. This method is an improved version of the Relief method and uses the Manhattan distance instead of the Euclid

**Table 3**
The parameters of MobileNetV2.

| Layer Name | Output Size | Filter size | Times |
|---|---|---|---|
| Conv2d | 224 × 224 × 3 | 32 | 1 |
| Bottleneck | 112 × 112 × 32 | 16 | 1 |
| Bottleneck | 112 × 112 × 16 | 16 | 2 |
| Bottleneck | 54 × 56 × 24 | 24 | 3 |
| Bottleneck | 28 × 28 × 32 | 32 | 4 |
| Bottleneck | 14 × 14 × 64 | 64 | 3 |
| Bottleneck | 14 × 14 × 96 | 160 | 3 |
| Bottleneck | 7 × 7 × 160 | 320 | 1 |
| Conv2d 1 × 1 | 7 × 7 × 320 | 1280 | 1 |
| Avgpool | 7 × 7 × 1280 | – | 1 |
| Conv2d 1 × 1 | 1 × 1 × 1280 | – | – |

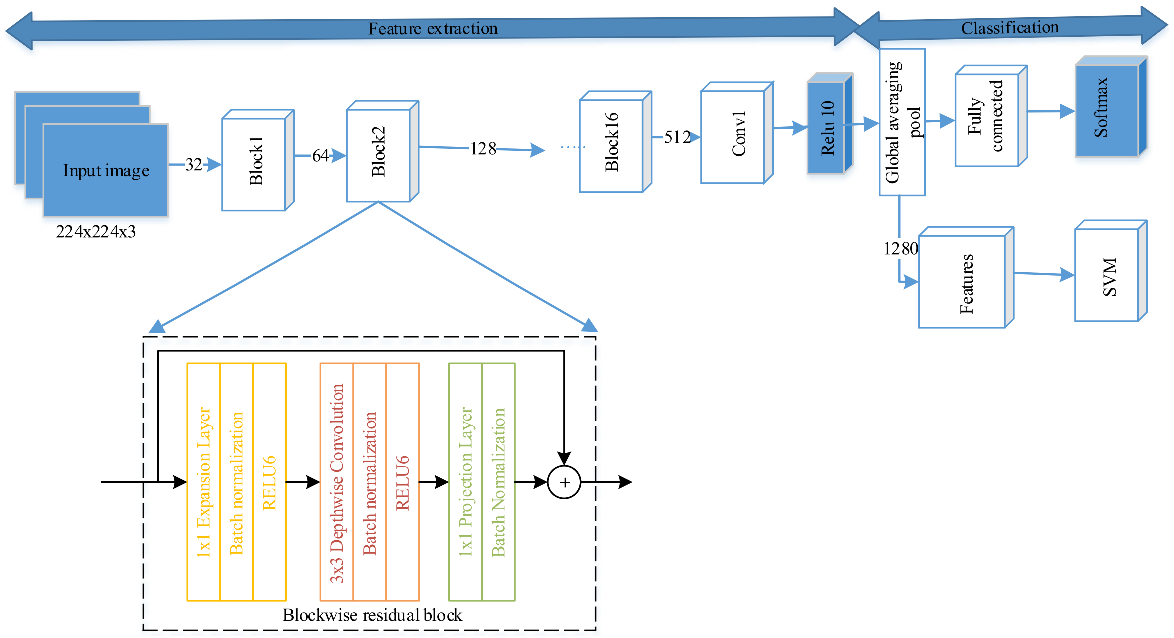


**Fig. 9.** MobileNetV2 architecture.

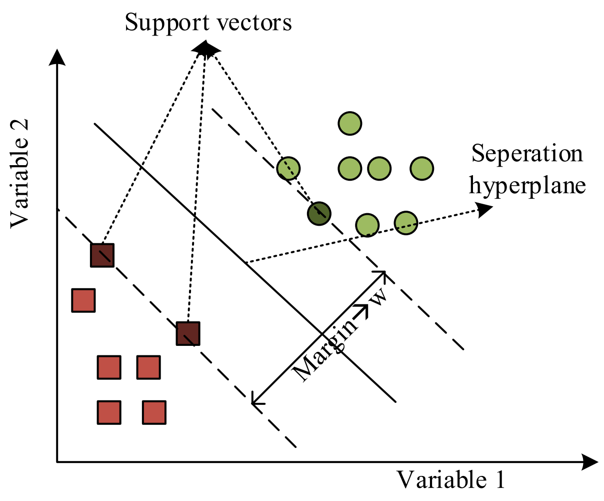


**Fig. 10.** Decision hyperplane of support vector machine.

distance to calculate the weights. This algorithm finds the weights of the estimators according to the nearest neighborhood value in a condition where there are multi-class categorical variables. The rank and weight of the predictors usually depends on the nearest neighbor number k. Initially, the weight values are chosen to be 0 for all estimators. The algorithm then chooses a random sample and updates the weights by finding the k-nearest neighbor. The weight update of each attribute is calculated in (4).

$$W[A] = W[A] - \sum_{j=1}^{k} \frac{d_A(R, H_j)}{m} + \sum_{c \neq class(R)} \left[\frac{P_Q}{(1-P_R)}\right] \sum_{j=1}^{k} \frac{d_A(R, M_j)}{m} \tag{4}$$

In (4), $W[A]$ represents the weight of an attribute A. R is a randomly selected sample, Hj is the closest sample with the same class of attribute A, and Mj is the closest sample with attribute A in a different class. The parameter m in (4) indicates the number of iterations, and k indicates the number of nearest neighbors. The $P_Q$ and $P_R$ values indicate the probability of the current sample and the randomly selected sample belonging to the class, respectively. The $d_A$ function shows the distance between two instances taken as parameters and is calculated as in (5).

$$d_A(X, Y) = \frac{|X - Y|}{A_{\max} - A_{\min}} \tag{5}$$

In (5), $A_{\max}$ and $A_{\min}$ represent the maximum and minimum values for the attribute $A$, respectively. The ReliefF algorithm iteratively updates the weights and ranks the attributes from the most weighted to the least in the classification. The reduced features obtained after the feature selection are given to the SVMs. SVM is a machine learning method used for classification and regression that constructs hyperplanes for classification purposes. For a good classification, the margin between the closest samples of different classes must be the maximum. Normally, SVMs are used for multi-class classification, although they are a two-class classification method. The one against the other or one against all methods is used for multi-class problems [32]. Fig. 10 shows the separation hyperplane in SVMs for linear separable samples.

The support vectors of the two classes are used to separate linear data, as shown in Fig. 10. Support vectors are the examples closest to the separation hyperplane. The aim is to maximize the margin between the closest instances of the two classes [33]. Selection of the best and most appropriate classification parameters of SVMs is actually seen as an optimization problem. Given the training samples as $x_i \in \Re^p$, $i = 1, \ldots n$, with class labels as $y \in \{-1 \quad 1\}$, the aim is to find the values of w and b, and we expect the $sign(w^T x + b)$ value to be correct for most examples. For this purpose, the following primal problem should be solved.

$$\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i \qquad \textit{subject to } y_i(w^T x_i + b) \geq 1 - \zeta_i, i = 1, \ldots, n \tag{6}$$

In (6), w and b represent the n-dimensional weight vector and bias value, respectively. The parameter $y_i$ takes a value of $-1$ or 1 as the class label. The margin is maximized by minimizing the $w^T w$ value. Ideally, the value of $y_i(w^T x_i + b)$ should always be $\geq 1$. However, some samples may be allowed to be at a border of their correct margin, since perfect classification cannot always be done. These variables are expressed as slack variables ($\zeta_i$). The factor determining the width of the margin is the penalty parameter C. Classification performance increases when this parameter is chosen appropriately by the user [34]. This is an optimization problem, and the values that will minimize the equation can be solved with Lagrange multipliers. Lagrange multipliers can be expressed as a quadratic programming problem and can be given as follows.

$$\begin{Bmatrix} \textit{Maximize} & L(\alpha) = \sum_{i=1}^{k} \alpha_i - \frac{1}{2} \sum_{i=0}^{k} \sum_{j=0}^{k} \alpha_i \alpha_j y_i y_j x_i x_j \\ \textit{Subject to} & \sum_{i=1}^{k} \alpha_i y_i = 0, \textit{for } i = 1 \ldots k \end{Bmatrix} \tag{7}$$

In (7), $\alpha_i$ is the Lagrange multipliers that take positive values between 0 and C. The value of $L(\alpha)$ is the Lagrange function that must be maximized for positive $\alpha_i$ values. For data that is a support vector in the data set, $\alpha_i$ values get positive values, while others take 0 values. The sequential minimization algorithm is used to solve this optimization problem [35]. If the problem cannot be solved in linear space, then the problem space is transformed into a nonlinear space using kernel functions. Popular kernel functions are polynomial, radial-based, and tanh-based kernel functions. In this study, SVMs will be used for the multi-class classification problem. Since the problem is nonlinear classification, the radial-based function will be used as a kernel function.

## 4. Experimental results

In order to evaluate the performance of the proposed method, images obtained by the Railways Research and Technology Center (DATEM) with the measurement train were used. Images were obtained from the travels made by the measurement train on the Ankara–Konya and Ankara–Eskisehir lines. A DalsaSpyder3 line scan camera with a resolution of 2096 pixels was used for image acquisition. The line rate of the camera is 68,000 lines per second and allows for a resolution of 1 mm × 0.15 mm during travel at a speed of 200 km per hour. A data set was formed by using healthy and different types of surface defects among the obtained images. Some examples from the data set are given in Table 4.

Some performance metrics were used to measure the accuracy and effectiveness of the proposed method. Metrics such as precision, recall, F1, and accuracy rate were obtained using four basic measurements: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). The TP value indicates the number of correctly classified samples within the full data set. The FP value is the number of instances estimated by a class other than its real class. The TN value is the number of samples whose class is negative and predicted as negative. The FN is the samples whose real class is negative and predicted as positive. The precision value (P) is the ratio of correctly predicted samples to the total number of samples. It is a measure of how much

**Table 4**
Railway surface defect data set.

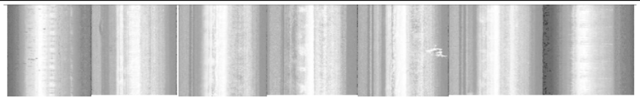
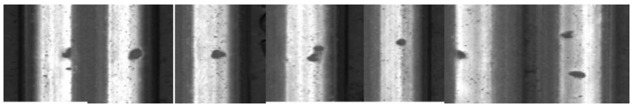
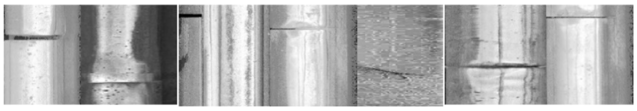
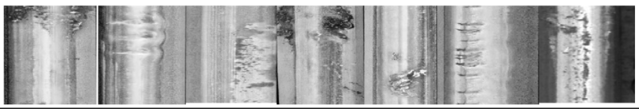
| Label | Example | Number of samples |
|---|---|---|
| Healthy |  | 492 |
| Light squat |  | 608 |
| Joint |  | 408 |
| Severe Squat |  | 330 |

**Table 5**
Comparisons of proposed ROI extraction method with other methods.

| Reference | Preprocessing | # correctly detected tracks | # incorrectly detected tracks | Accuracy rate (%) |
|---|---|---|---|---|
| [10] | Histogram equalization | 170 | 30 | 85.00 |
| [12] | Michelson-Like contrast enhancement | 180 | 25 | 87.50 |
| [24] | Median filter and segmentation | 185 | 15 | 92.50 |
| Proposed method | High contrast image | **195** | **5** | **97.50** |

**Table 6**
The training parameters of SqueezeNet and MobileNetV2.

| Parameter | Value |
|---|---|
| Input image size | 224 × 224 |
| Optimization | sgdm |
| Momentum | 0.9 |
| Learning rate | 1e−4 |
| Mini Batch size | 64 |
| Max epoch | 10 |

**Table 7**
Classification results for each deep learning model.

| Classification Accuracy | | | Computation time | |
|---|---|---|---|---|
| Model | Training | Validation | Train | Test |
| SqueezeNet | 97.44 | 90.38 | 93 s | 21 ms |
| MobileNetv2 | 96.04 | 94.37 | 67 s | 15 ms |

is correctly predicted and takes a value between 0 and 1 and should be as high as possible. The recall value (R) represents how successfully the positive samples are predicted and is a measure of the accuracy of the tested data. F1 is the harmonic mean of the precision and recall metrics. The performance metrics for Accuracy ($A$), $P$, $R$, and $F1$ are defined as follows.

$$A = \frac{TP + TN}{TP + TN + FP + FN} \tag{8}$$

$$P = \frac{TP}{TP + FP} \tag{9}$$

$$R = \frac{TP}{TP + FN} \tag{10}$$

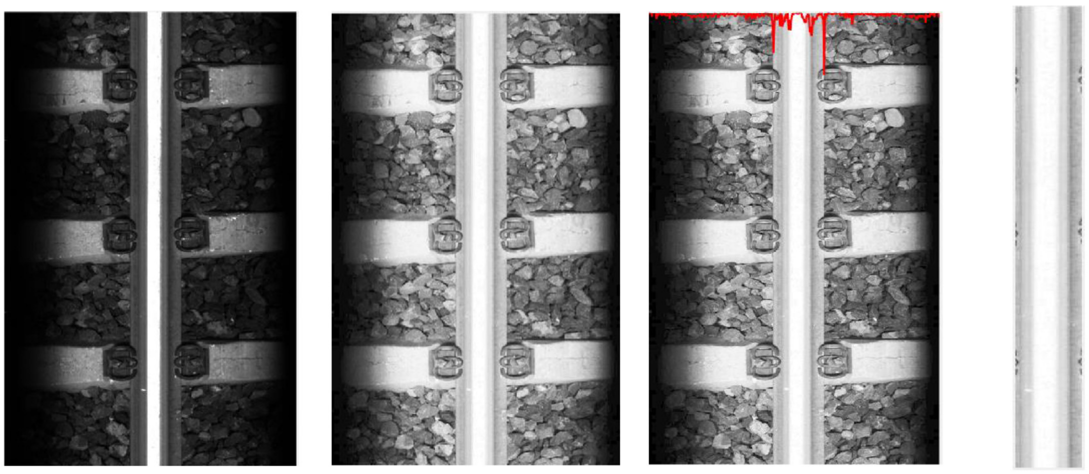
$$F1 = 2x\frac{PXR}{P + R} \tag{11}$$

Image processing and defect detection algorithms were implemented in the MATLAB environment using image processing and deep learning toolboxes. A workstation computer (CPU: Intel i7, GPU: GTX1080, 8GB, RAM: 16GB) was used to run the programs. To detect the rail surface defects, it is necessary to first crop the region of interest from the image and then construct the data set. The proposed method first obtains a high contrast image by applying a contrast enhancement method. The histogram-based technique is then applied to extract the region of interest from this image. In Fig. 11, the track extraction from the rail image is shown step by step.

In the original track image shown in Fig. 11(a), some parts appear brighter due to the lighting, while other parts are darker. Fig. 11(b) shows that these problems were solved after the image enhancement. In the ROI extraction step, the two maximum values are found from the signal obtained in Fig. 11(c). The cropping rail part according to this selected region is shown in Fig. 11(d). The proposed rail extraction method has been compared with the methods in the literature. For this purpose, 200 images were randomly selected from the data set for comparison. The algorithm automatically reads the images in the folder, extracts the rail region, and writes the cropping images to a different folder. The detection accuracy of the method was calculated as the number of correctly and incorrectly detected tracks. The comparison results for this purpose are given in Table 5.

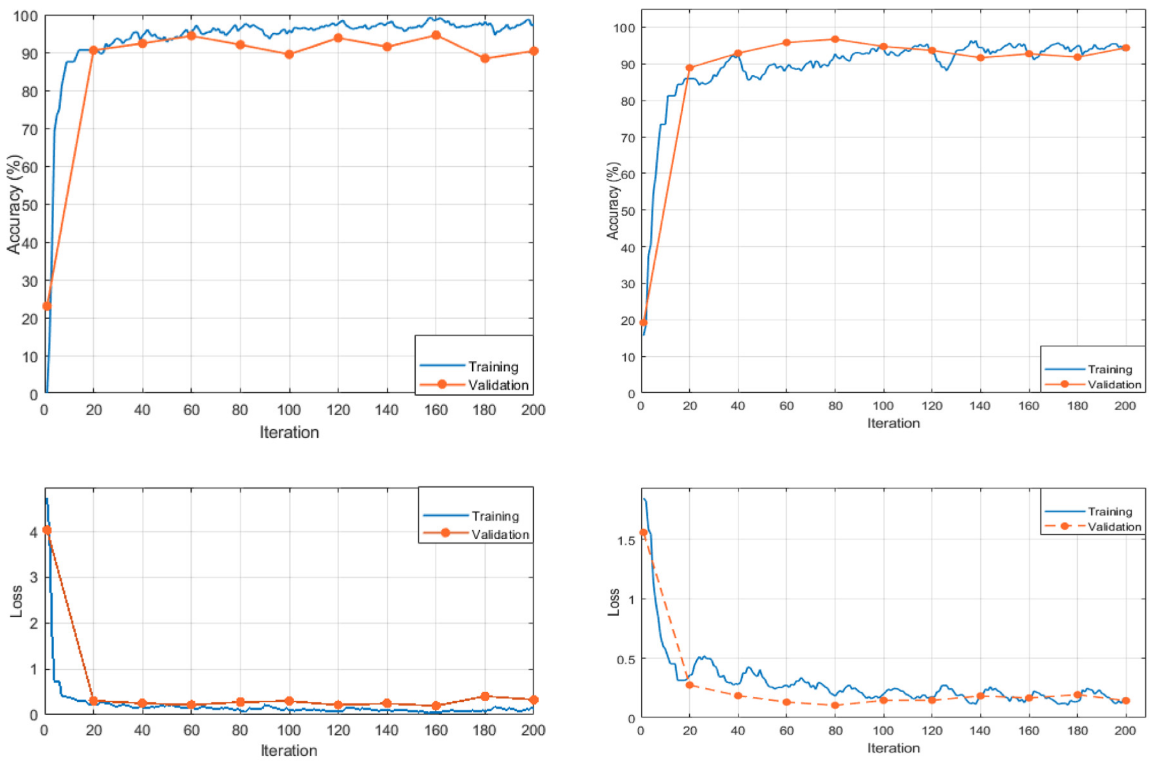
According to the results given in Table 5, it is seen that the proposed method gives better results than other methods. In [10] and [12], only the starting point of the rail is determined, and the width of the rail should be given as a parameter to the algorithm. In [22], problems related to illumination cannot be solved, since only 3 × 3 size median filters are used in the preprocessing step. After the track region is obtained from the original image, the data set should be constructed and the model trained for classification. The first constructed data set was trained with pre-trained deep learning models, such as SqueezeNet and MobileNetV2 using transfer learning. The training parameters of the deep learning models are given in Table 6.

After determining the training parameters given in Table 6, the constructed deep learning models were trained with the given data set. During the training stage, 70% of the data set was used in training, and 30% was used for validation. The constructed deep learning model ran on 200 iteration steps. Fig. 12 shows the



(a) Original image (b) Image enhancement (c) Region of interest extraction (d) Cropped image

**Fig. 11.** Track position extraction.



(a) Accuracy and loss graphs of SqueezeNet **(b)** Accuracy and loss graphs of MobileNetv2

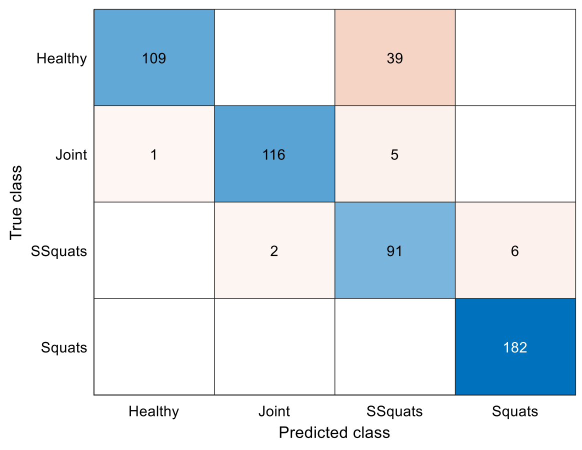
**Fig. 12.** The training progress of deep learning models.

progress of training and validation achievements for each deep learning model.

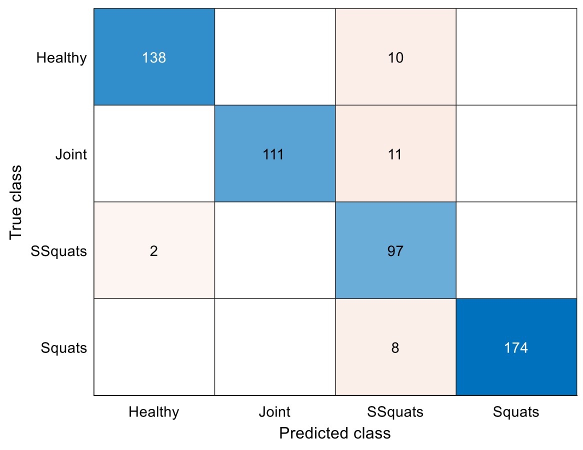
In Fig. 12(a), although the training performance of the SqueezeNet deep learning model is high, the validation performance is low. In Fig. 12(b), the training and validation performances were obtained closer to each other. The performance and calculation times of the two deep learning models are given in Table 7.

As shown in Table 7, although the difference between the training accuracies of two models is low, the accuracy rate of MobileNetV2 is higher in the validation data set. In addition, the MobileNetV2 process was completed in a shorter amount of time in terms of calculation time. Fig. 13 represents the confusion matrices of the two models to clearly illustrate the classification rate of each category.

In Fig. 13, it can be seen from the confusion matrix that the MobileNetV2-based method for the validation process achieves better classification accuracy than SqueezeNet. In the SqueezeNet model, it is seen that the SSquat and Healthy cases were confused. The model proposed in this study extracts features from each

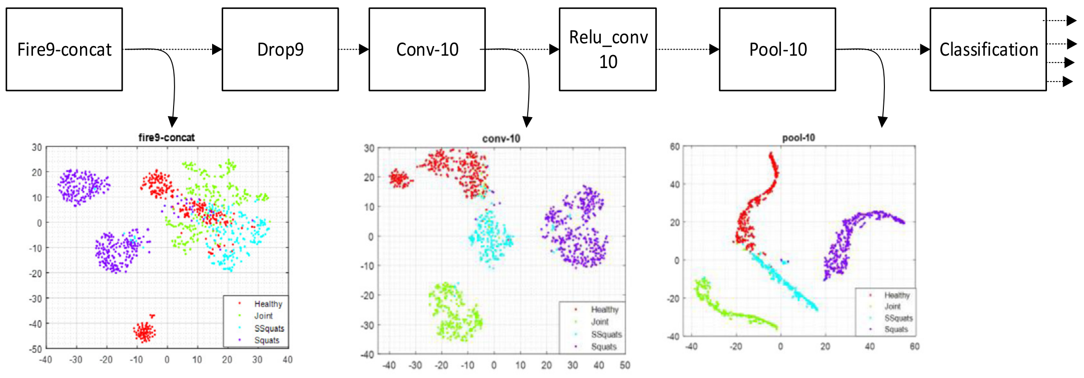


(a) SqueezeNet

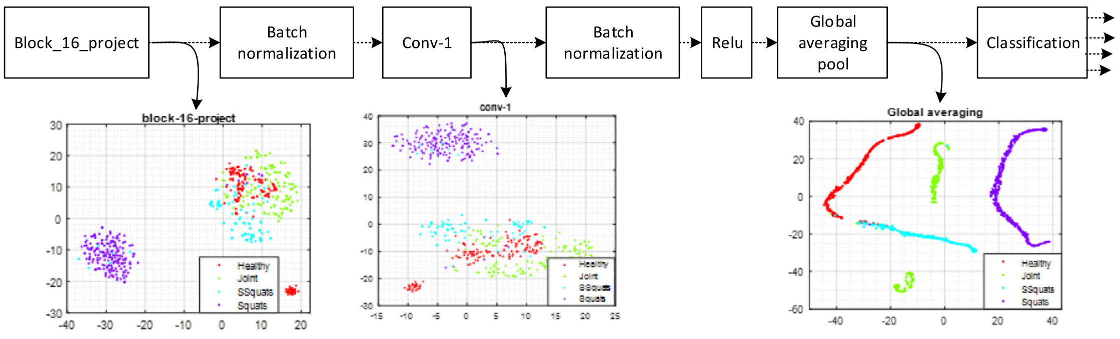


(b) MobileNetv2

**Fig. 13.** Confusion matrix of two deep learning models for validation data set.



(a) SqueezeNet



(b) MobileNetv2

**Fig. 14.** Visualization of extracted features of each deep learning models for different layers.

deep neural network and classifies these features with SVMs after feature reduction. Fig. 14 shows the features of each model obtained in different layers.

The features obtained from the pool10 layer of SqueezeNet and the global averaging pool of MobileNetV2 are given to the ReliefF algorithm, which calculates the feature weights. In Fig. 15, the importance order of the features with the ReliefF algorithm is given for the 500 features obtained for each deep learning model.

In Fig. 15, the weights of the first 500 features with high importance for each deep learning model are given. During the feature extraction, 1,000 and 1,287 features were obtained in SqueezeNet and MobileNetV2, respectively. As a result of the experiments, it was seen that the selection of 500 features was sufficient. It was observed that as the number of selected features decreased, the classification performance decreased, but with values higher than 500 features, the performance did not change. The nearest neighbor number of the ReliefF algorithm was chosen as 10. In Fig. 16(a) and (b), the confusion matrices obtained in the classification of the features obtained from SqueezeNet and MobileNetV2 deep learning models with SVMs are given, respectively. In Fig. 16(c), the feature selection is made with the ReliefF algorithm by fusing the features of the two models. Then, the confusion matrix obtained by classifying these features with SVMs is given for the validation data set.

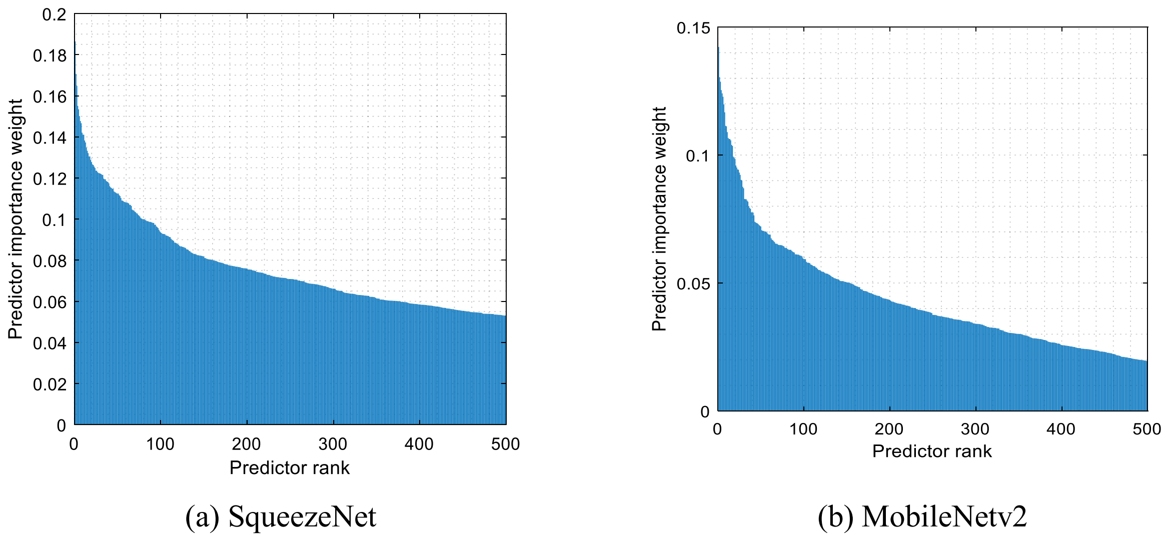
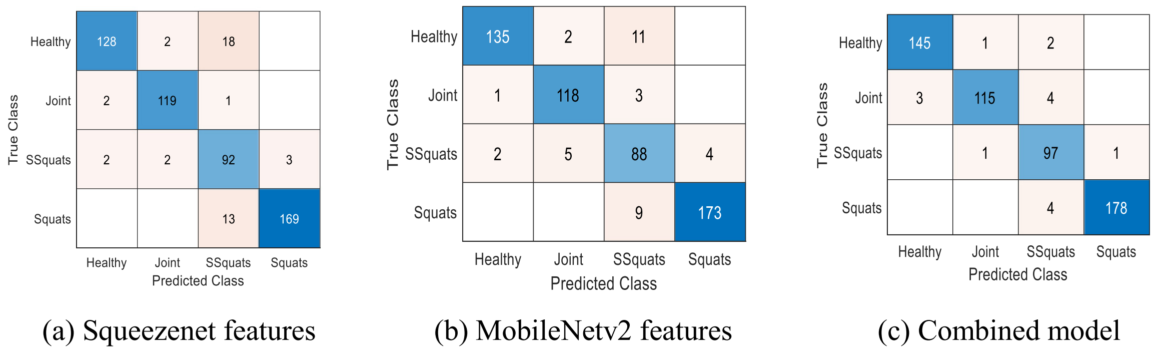


**Fig. 15.** The importance order of features obtained from each deep learning model.



(a) Squeezenet features

| True \ Predicted | Healthy | Joint | SSquats | Squats |
|---|---|---|---|---|
| Healthy | 128 | 2 | 18 | |
| Joint | 2 | 119 | 1 | |
| SSquats | 2 | 2 | 92 | 3 |
| Squats | | | 13 | 169 |

(b) MobileNetv2 features

| True \ Predicted | Healthy | Joint | SSquats | Squats |
|---|---|---|---|---|
| Healthy | 135 | 2 | 11 | |
| Joint | 1 | 118 | 3 | |
| SSquats | 2 | 5 | 88 | 4 |
| Squats | | | 9 | 173 |

(c) Combined model

| True \ Predicted | Healthy | Joint | SSquats | Squats |
|---|---|---|---|---|
| Healthy | 145 | 1 | 2 | |
| Joint | 3 | 115 | 4 | |
| SSquats | | 1 | 97 | 1 |
| Squats | | | 4 | 178 |

**Fig. 16.** Confusion matrices of each model by using feature extraction and SVMs.

When the confusion matrices in Fig. 16 are examined, it is shown that the feature fusion has a better performance than using a single model. In both SqueezeNet and MobileNetV2, it is seen that the SSquat class is confused with healthy class instances, and the classification performance is low. The performance of the proposed method was compared with other deep learning models according to accuracy, precision, recall, and F1 metrics. The comparison results are given in Table 8.

In Table 8, the comparison results of the proposed approach with different deep learning models are given according to four metrics. In addition, the feature extraction layer and the number of features in the relevant layer are given. Especially in the SSquats class, it was observed that the performance metrics of the other methods are low. All the performance metrics of our method were over 90%. In addition, fewer features are used in the test phase, and better performance was achieved compared to other deep learning models. Approaches that exist in the literature for determining the rail surface defects are generally based on determining if there are defects. Therefore, methods in the literature generally divide the rail surface into two classes, namely healthy and defect. In only one study, the defect types were determined with a laser-based three-dimensional approach [22]. Finally, we aimed to determine the contribution of the proposed method to the literature by comparing the error classification performances between the presented deep learning-based integrated method and the methods available in the literature. The comparison results are given in Table 9.

The results in Table 9 prove that by combining the features obtained with the two deep learning models our method achieves better classification accuracy than other rail defect detection methods. The method in [1] first enhances the image with gray equalization and extracts the region of interest. Then, the noise is reduced with a curvature filter, and the image is segmented with a Gaussian mixture model. In [3], defects are determined by applying an entropy-based method to the rail surface images after contrast enhancement. In [14], after the image is enhanced, the possible defect points are determined by processing the image line by line. In [18], a segmented edge extraction-based method was proposed to reduce the problems associated with lighting and to detect rail defects with high accuracy. However, it was stated that the proposed method in that paper was insufficient to detect different types of defects. The study proposed in [19] proposes a multiple defect detection method based on image processing and deep learning that was tested on different rail surfaces. First, the defect area was determined by segmentation, and defect detection was made with deep learning. However, their proposed method has disadvantages in that it is both complex and does not determine the type of defect. Most of the studies in the literature have focused only on defect detection on the rail surface. Therefore, almost no study has been done to determine the type of defect. The suggested studies usually distinguished between segmentation-based defects and the rail surface. Traditional segmentation methods are highly affected by lighting problems and noise such as dirt and dust on the rail surface, so various methods have been proposed to overcome these problems. However, the flaw detection performance of these methods remained low. The approach proposed in [22] obtains a 3-dimensional image by using a laser camera, creates

**Table 8**
Performance comparison results of different deep learning models.

| Model | Class | Layer | Features | P | R | F1 | Accuracy (%) |
|---|---|---|---|---|---|---|---|
| SqueezeNet | Healthy | Pool10 | 1000 | 0.96 | 0.86 | 0.91 | 92.45 |
| | Joint | | | 0.96 | 0.97 | 0.97 | |
| | SSquats | | | 0.74 | 0.92 | 0.82 | |
| | Squat | | | 0.98 | 0.92 | 0.95 | |
| MobileNetv2 | Healthy | Global average pool | 1287 | 0.97 | 0.91 | 0.94 | 93.28 |
| | Joint | | | 0.94 | 0.96 | 0.95 | |
| | SSquats | | | 0.79 | 0.88 | 0.83 | |
| | Squat | | | 0.97 | 0.95 | 0.96 | |
| GoogleNet | Healthy | Pool5 drop | 1024 | 0.92 | 0.97 | 0.95 | 94.90 |
| | Joint | | | 0.97 | 0.95 | 0.96 | |
| | SSquats | | | 0.91 | 0.82 | 0.86 | |
| | Squat | | | 0.97 | 0.98 | 0.98 | |
| VGG19 | Healthy | Drop7 | 4096 | 0.93 | 0.96 | 0.95 | 95.30 |
| | Joint | | | 0.96 | 0.95 | 0.96 | |
| | SSquats | | | 0.90 | 0.86 | 0.88 | |
| | Squat | | | 0.97 | 0.98 | 0.98 | |
| Resnet50 | Healthy | Avg-pool | 2048 | 0.94 | 0.97 | 0.96 | 95.40 |
| | Joint | | | 0.95 | 0.97 | 0.96 | |
| | SSquats | | | 0.94 | 0.83 | 0.88 | |
| | Squat | | | 0.96 | 0.98 | 0.97 | |
| The Proposed Model | Healthy | Concatenated of two models | **1000** | **0.97** | **0.97** | **0.97** | **97.10** |
| | Joint | | | **0.98** | **0.94** | **0.96** | |
| | SSquats | | | **0.90** | **0.97** | **0.94** | |
| | Squat | | | **0.99** | **0.97** | **0.98** | |

**Table 9**
Comparison of performance results with other methods.

| Reference | Used method | Number of class | Accuracy rate (%) |
|---|---|---|---|
| [1] | Gaussian mixture models and curvature filter based segmentation | 2 | 95.64 |
| [3] | Emphasized maximum entropy based segmentation | 2 | 90.07 |
| [14] | Coarse to fine model and otsu based thresholding | 2 | 88.53 |
| [18] | Edge detection and contour based region growing | 2 | 92.03 |
| [19] | Image processing and deep learning based multi-model learning | 2 | 94.70 |
| [22] | Geometrical features and neural networks based classifier | 6 | 93.30 |
| Our method | Combined features of two deep learning methods and SVMs | **4** | **97.10** |

a point cloud, and classifies the obtained geometric features from the cloud point with an artificial neural network. However, it takes a lot of time to obtain the 3-dimensional image and to process it in real time. Therefore, the experiments were carried out in the laboratory environment. Unlike the other methods in the literature, our proposed method determines the types of defects according to the features extracted through deep neural networks instead of segmentation after determining the rail area. Thus, higher performance was achieved, as the rail surface is better modeled.

## 5. Conclusions

This paper concerned the detection and classification of rail surface defects. The proposed method consists of image processing, feature extraction from multiple deep learning models, feature combining, and classification with SVMs. The proposed image processing-based approach obtains a high contrast image from the original image, and the region of the rail surface is cropped. The proposed contrast enhancement approach eliminates the problems that occur on the rail surface due to illumination and increases the performance of the ROI extraction method. After the rail track was obtained, feature extraction, feature combination from multiple deep learning networks, feature selection, and classification with SVMs were applied in order to classify the surface defects. The SqueezeNet and MobileNetV2 models were used for feature extraction, and the ReliefF algorithm was used for feature selection. The proposed approach achieved a classification success of 97.10%, which is better than the reported methods available in the literature. To the best knowledge of the authors, this method was the first method to use deep learning models to detect rail surface defects.

The proposed method in the paper is demonstrated by images collected from the Ankara–Konya and Ankara–Eskisehir lines of Turkey Republic state railways. The obtained images provide some alternatives for practical applications. These methods will be used in field tests in the near future as part of our work on this project.

## Declaration of competing interest

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

## Acknowledgment

This work was supported by the TUBITAK (The Scientific and Technological Research Council of Turkey) under Grant No: 120E097. The data set created have been uploaded to GitHub. You can access the code and data set from the following link: https://github.com/ilhanaydintr/RailDefectDetection.

## References

[1] H. Zhang, X. Jin, Q.J. Wu, Y. Wang, Z. He, Y. Yang, Automatic visual detection system of railway surface defects with curvature filter and improved Gaussian mixture model, IEEE Trans. Instrum. Meas. 67 (7) (2018) 1593–1608, https://doi.org/10.1109/TIM.2018.2803830.

[2] F. Espinosa, J.J. García, A. Hernández, M. Mazo, J. Urena, J.A. Jimenez, J.C. Garcia, Advanced monitoring of rail breakage in double-track railway lines by means of PCA techniques, Appl. Soft Comput. 63 (2018) 1–13, https://doi.org/10.1016/j.asoc.2017.11.009.

[3] Q. Li, S. Ren, A visual detection system for rail surface defects, IEEE Trans. Syst. Man Cybern. C 42 (6) (2010) 1531–1542, https://doi.org/10.1109/TSMCC.2012.2198814.

[4] M.B. Kishore, J.W. Park, S.J. Song, H.J. Kim, S.G. Kwon, Characterization of defects on rail surface using eddy current technique, J. Mech. Sci. Technol. 33 (9) (2019) 4209–4215, https://doi.org/10.1007/s12206-019-0816-x.

[5] J. Zhang, H. Ma, W. Yan, Z. Li, Defect detection and location in switch rails by acoustic emission and lamb wave analysis: A feasibility study, Appl. Acoust. 105 (2016) 67–74, https://doi.org/10.1016/j.apacoust.2015.11.018.

[6] A.A. Markov, E.A. Maksimova, A.G. Antipov, Analyzing the development of rail defects based on results of multichannel periodic testing, Russian J. Nondestruct. Test. 55 (2019) 875–886, https://doi.org/10.1134/S1061830919120064.

[7] Y. Ou, J. Luo, B. Li, B. He, A classification model of railway fasteners based on computer vision, Neural Comput. Appl. 31 (2019) 9307–9319, https://doi.org/10.1007/s00521-019-04337-z.

[8] H. Fan, P.C. Cosman, Y. Hou, B. Li, High-speed railway fastener detection based on a line local binary pattern, IEEE Signal Process. Lett. 25 (2018) 788–792, https://doi.org/10.1109/LSP.2018.2825947.

[9] J. Liu, Y. Huang, Q. Zou, M. Tian, S. Wang, X. Zhao, S. Ren, Learning visual similarity for inspecting defective railway fasteners, IEEE Sens. J. 19 (2019) 6844–6857, https://doi.org/10.1109/JSEN.2019.2911015.

[10] X. Wei, Z. Yang, Y. Liu, D. Wei, L. Jia, Y. Li, Railway track fastener defect detection based on image processing and deep learning techniques: A comparative study, Eng. Appl. Artif. Intell. 80 (2019) 66–81, https://doi.org/10.1016/j.engappai.2019.01.008.

[11] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, IEEE Trans. Pattern Anal. Mach. Intell. 39 (2017) 1137–1149, https://doi.org/10.1109/TPAMI.2016.2577031.

[12] Y. Wu, Y. Qin, Z. Wang, L. Jia, A UAV-based visual inspection method for rail surface defects, Appl. Sci. 8 (2018) 1–20, https://doi.org/10.3390/app8071028.

[13] Z. Zheng, H. Qi, L. Zhuang, Z. Zhang, Automated rail surface crack analytics using deep data-driven models and transfer learning, Sustainable Cities Soc. 70 (2021) 1–11, https://doi.org/10.1016/j.scs.2021.102898.

[14] Q. Li, S. Ren, A visual detection system for rail surface defects, IEEE Trans. Syst. Man Cybern. C 42 (2012) 1531–1542, https://doi.org/10.1109/TSMCC.2012.2198814.

[15] H. Yu, Q. Li, Y. Tan, J. Gan, J. Wang, Y.A. Geng, L. Jia, A coarse-to-fine model for rail surface defect detection, IEEE Trans. Instrum. Meas. 68 (2019) 656–666, https://doi.org/10.1109/TIM.2018.2853958.

[16] J. Gan, J. Wang, H. Yu, Q. Li, Z. Shi, Online rail surface inspection utilizing spatial consistency and continuity, IEEE Trans. Syst. Man Cybern. Syst. 50 (2020) 2741–2751, https://doi.org/10.1109/TSMC.2018.2827937.

[17] M. Nieniewski, Morphological detection and extraction of rail surface defects, IEEE Trans. Instrum. Meas. 69 (2020) 6870–6879, https://doi.org/10.1109/TIM.2020.2975454.

[18] X. Ni, H. Liu, Z. Ma, C. Wang, J. Liu, Detection for rail surface defects via partitioned edge feature, IEEE Trans. Intell. Transp. Syst. (2021) (2021) 1–17, https://doi.org/10.1109/TITS.2021.3058635.

[19] X. Jin, Y. Wang, H. Zhang, H. Zhong, L. Liu, Q.J. Wu, Y. Yang, DM-RIS: Deep multimodel rail inspection system with improved MRF-GMM and CNN, IEEE Trans. Instrum. Meas. 69 (2020) 1051–1065, https://doi.org/10.1109/TIM.2019.2909940.

[20] C. Taştimur, M. Karaköse, E. Akın, E.I. Aydın, Rail defect detection with real time image processing technique, in: 2016 IEEE 14th International Conference on Industrial Informatics, INDIN, 2016, pp. 411415.

[21] G. Karaduman, M. Karakose, I. Aydin, E. Akin, Contactless rail profile measurement and rail fault diagnosis approach using featured pixel counting, Intell. Autom. Soft Comput. 26 (2020) 455–463, https://doi.org/10.32604/iasc.2020.013922.

[22] J. Ye, E. Stewart, D. Zhang, Q. Chen, C. Roberts, Method for automatic railway track surface defect classification and evaluation using a laser-based 3D model, IET Image Process. 14 (2020) 2701–2710, https://doi.org/10.1049/iet-ipr.2019.1616.

[23] J. Xie, C. Huang, C. Zeng, S.H. Jiang, N. Podlich, Systematic literature review on data-driven models for predictive maintenance of railway track: implications in geotechnical engineering, Geosciences 10 (2020) (2020) 1–24, https://doi.org/10.3390/geosciences10110425.

[24] Y. Min, B. Xiao, J. Dang, B. Yue, T. Cheng, Real time detection system for rail surface defects based on machine vision, EURASIP J. Image Video Process. 1 (2018) (2018) 1–11, https://doi.org/10.1186/s13640-017-0241-y.

[25] MathWorks, Image Processing Toolbox for MATLAB 2021a, The MathWorks Inc., 2021, (Accessed 15 April 2021).

[26] R.C. Gonzalez, R.E. Woods, S.L. Eddins, Digital Image Processing using MATLAB, Pearson Education India, 2004.

[27] D. Mery, F. Pedreschi, Segmentation of colour food images using a robust algorithm, J. Food Eng. 66 (2005) 353–360, https://doi.org/10.1016/j.jfoodeng.2004.04.001.

[28] J.C. Lagarias, J.A. Reeds, M.H. Wright, P.E. Wright, Convergence properties of the nelder-mead simplex method in low dimensions, SIAM J. Optim. 19 (1998) 112–147, https://doi.org/10.1137/S1052623496303470.

[29] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.C. Chen, Mobilenetv2: Inverted residuals and linear bottlenecks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4510–4520.

[30] F.N. Iandola, S. Han, M.W. Moskewicz, K. Ashraf, W.J. Dally, K. Keutzer, Squeezenet: Alexnet-level accuracy with 50x fewer parameters and < 0.5 MB model size, 2016, arXiv preprint arXiv:1602.07360.

[31] M.R. Sikonja, I. Kononenko, Theoretical and empirical analysis of relieff and rrelieff, Mach. Learn. 53 (2003) 23–69, https://doi.org/10.1023/A:1025667309714.

[32] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, second ed., Springer, New York, 2008.

[33] L. Tang, Y. Tian, W. Li, P.M. Pardalos, Structural improved regular simplex support vector machine for multiclass classification, Appl. Soft Comput. 91 (2020) 106235, https://doi.org/10.1016/j.asoc.2020.106235.

[34] I. Aydin, M. Karakose, E. Akin, A multi-objective artificial immune algorithm for parameter optimization in support vector machine, Appl. Soft Comput. 11 (2011) 120–129, https://doi.org/10.1016/j.asoc.2009.11.003.

[35] J. Platt, Chapter fast training of support vector machines using sequential minimal optimization, in: Advances in Kernel Methods-Support Vector Learning, vol. 36, MIT Press, 1999, pp. 185–208.