International Journal on Advanced Science Engineering Information Technology

Vol.14 (2024) No. 5
ISSN: 2088-5334

Design of Cost-Effective Steel Surface Defect Detector Based on Deep Learning

Jundong Lee [a], Hwan Seog Kim [b], Yong Wan Ju [c], Byoungwook Kim [d], Sangmin Suh [b,*]

[a] *Department of Multimedia Engineering, University, Gangneung-Wonju National University, Wonju, Republic of Korea*
[b] *Department of Information and Telecommunication Engineering, Gangneung-Wonju National University, Wonju, Republic of Korea*
[c] *Industrial Academic Cooperation Group, Gangneung-Wonju National University, Wonju, Republic of Korea*
[d] *Department of Computer Science and Engineering, Gangneung-Wonju National University, Wonju, Republic of Korea*
*Corresponding author: *sangminsuh@gwnu.ac.kr*

Abstract—**Considering the productivity and profits of manufacturers, defects in steel products must be detected very carefully. Traditionally, machine learning-based methods such as support vector machines (SVM) have been widely used for steel surface defect detection (SDD). These machine learning methods rely on expert parameters, so their performance is susceptible to these parameters. Recently, deep learning-based methods independent of these expert parameters have been developed. The most popular of these uses is transfer learning, which allows for efficient learning by applying a previously trained model to a new problem. This approach can overcome existing methods' limitations and provide more accurate and efficient SDD solutions. In order to design the transfer learning-based SDD, input images should be modified and resized, which could cause input image distortion. Moreover, the transfer learning-based models generally have many layers and weights. This paper is motivated by the following questions. 1) Is the transfer learning-based model the best method for SDD? 2) What is the smallest model without compromising performance? This paper proposes a dedicated neural network for steel surface defect detection to answer the above questions. In addition, for cheap neural networks, the initially designed neural network is gradually reduced by monitoring the performances. We achieved a maximum f1-score of 0.978 and a minimum AUC of 0.995 from the experimental results.**

Keywords— **Artificial intelligence; convolutional neural network; deep learning; steel surface defect detector; inference time.**

Manuscript received 5 Dec. 2023; revised 7 Apr. 2024; accepted 12 Sep. 2024 . Date of publication 31 Oct. 2024.

IJASEIT is licensed under a Creative Commons Attribution-Share Alike 4.0 International License.

I. INTRODUCTION

Productivity and profitability in steel manufacturing require proactive product defect detection [1]–[3]. However, accurate and fast defect detection is not accessible due to the reflective nature of the material, the poor manufacturing environment, and foreign substances such as dust, iron powder, oil, and water [4]–[6]. Surface defect detection can be accomplished using magnetic flux leakage (MFL) techniques and imaging. MFL is a specialized nondestructive testing (NDT) method that converts magnetic flux leaking from a defect into an electrical signal when a ferromagnetic test piece is magnetized, taking advantage of the fact that the deeper the defect, the stronger the electrical signal [7],[8]. Traditionally, image processing using machine learning has been the primary way to use images [9]–[10]. A steel defect classifier using HARR classifier and support vector machine (SVM) was also reported [11]–[13], and a method was proposed to determine whether the image to be inspected is defective by comparing the normal image and the image to be inspected and then analyzing the statistical value [14]–[16].

Recently, deep learning has been used in various fields to design fire alarm systems, such as adding a new smoke class for early fire warning [17]–[19], and in eye disease diagnosis, a system to distinguish between six diseases has been proposed [20]. These artificial intelligence neural networks have also been used to detect surface defects in steel. However, the proposed method utilizes transfer learning. Transfer learning is a method that borrows the model structure and trained weights from the winning ImageNet Large Scale Visual Recognition Challenge (ILSVRC) and modifies them to fit the purpose of using only the input and output parts of the neural network.

Since there are 1,000 classes to be distinguished in ILSVRC, the model used in transfer learning is necessarily deep and wide in the layers of the neural network [21], [22]. This is because the model requires a large multiplier and accumulator (MAC) and a large amount of memory to store many weights, so models used for transfer learning will inevitably only work on expensive systems [23]. ResNet, a popular model for transfer learning, has a structure that utilizes residual error to speed up training and is arguably an outstanding performer [24]. ResNet152, with 152 layers, has many layers and correspondingly large weights and was used to win the ILSVRC in 2015. A steel surface defect detector using ResNet152 was designed [25], and a detector using ResNet50 with a reduced model size was proposed [26]. The original model, ResNet152, and even the reduced-size ResNet50, have inherently large layers and weights, requiring expensive, high-performance systems. Attempts have been made to use MobileNet to create a more cost-competitive model [27]–[29]. MobileNet significantly reduces model size for use on mobile devices [30]. More recently, a method not using transfer learning has been proposed to substantially reduce the model's size with a slight performance penalty [31], [32].

In this paper, we propose a steel surface detector that optimizes the previously proposed model to reduce its size to a minimum while achieving better performance. To detect defects on the steel surface, it is necessary to be able to detect the texture of the surface well, so we designed a basic model based on a convolutional neural network (CNN). After developing the basic model, we iteratively reduce the layers while checking the performance to reduce the model gradually. The minimum-sized model obtained was evaluated on several performance metrics for fairness. In addition, inference time is also measured and compared to verify the effectiveness of model reduction.

This paper is organized as follows. In Section 2, the proposed optimized steel surface defect detector is designed and trained; Section 3 uses the designed neural network to conduct experiments with real images and evaluate its performance; and Section 4 presents conclusions.

II. Materials and Method

This chapter details the theoretical background, description of the dataset used, model design method, and model training method for designing a steel surface detector based on artificial neural networks.

A. Theoretical Background

The CNN-based surface defect detector is roughly divided into image feature extraction and image classification stages, with the overall structure shown in Fig. 1.

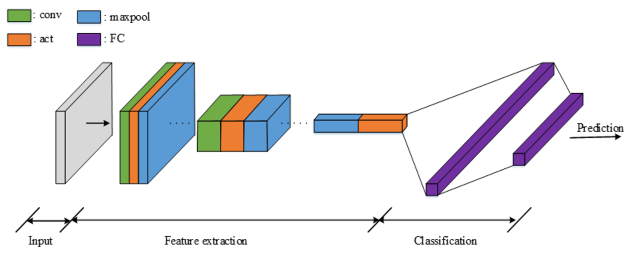


Fig. 1 Basic structure of SDD

The feature extraction stage extracts the texture characteristics of the input image and passes the results to the image classifier stage. To extract an image's texture, we use CNN. The layers of a CNN are convolution, activation function, and pooling. Convolution abstracts and quantizes the input image's information, and the kernel's size determines how much area of the image is of interest, which is defined as the receptive field. The behavior of convolution is represented by the equation (1).

$$\begin{aligned} s(j,i) &= z(j,i) \odot u \\ &= \sum_{x=-(h-1)/2}^{(h-1)/2} \sum_{y=-(h-1)/2}^{(h-1)/2} z(j+y,\, i+x)\, u(y,x) \end{aligned} \tag{1}$$

where *z*, *s* and *u* are the inputs, outputs and kernel. *h* is the size of the kernel, which determines the size of the receptive field. Typically, *h* is an odd number to ensure the output abstraction is centered on the input image. The activation function is a rectifier linear unit (ReLU). The ReLU was proposed to improve the vanishing gradient problem of sigmoid functions and is expressed as follows.

$$s(j,i) = \max(0, z(j,i)) \tag{2}$$

The pooling layer reduces the computation of the next layer and makes the model more robust by making it insensitive to noise. In this paper, we use max-pooling, which is represented as follows:

$$s(j,i) = \max_{x=0,\dots,h,\, y=0,\dots,h} z(j+y, i+x) \tag{3}$$

In Eq. (3), *h* is the stride, which determines how much to move by.

In the image classification stage, a layer, also called multilayer perceptron (MLP) or fully connected layer (FC), is used to classify the extracted features. The fully connected layer is represented as follows:

$$s = f(Wz + b) \tag{4}$$

where $s \in R^{m\times1}$, $W \in R^{m\times n}$, $z \in R^{n\times1}$, $b \in R^{m\times1}$. The input is *z*, the output is *s*, and *b* is a bias vector with the same elements. This fully connected layer can also be used in the hidden layer and the final layer, where the function *f* uses the ReLU in equation (2). The final layer uses SoftMax, which converts the hidden layer values to probabilities, such as in equation (5).

$$s_j = \frac{e^{z_j}}{\sum_{i=1}^{C} e^{z_i}} \tag{5}$$

where *C* is the number of classes to be classified and z_j is the *j*-th input. Since SoftMax converts inputs to probabilities, the following expression is guaranteed.

$$\sum_{i=1}^{C} e^{z_i} = 1 \tag{6}$$

Finally, the best probabilistic values are selected and exported as the final output.

$$s = \underset{j=1,\dots,C}{\arg\max}(e_j^z) \tag{7}$$

B. Datasets

The paper uses a dataset from Northeastern University [33]. The dataset is categorized into six different defects in sheet steel.

TABLE I
DEFECT CLASSES

Number	Class Name
0	Crazing
1	Inclusion
2	Patches
3	Pitted surface
4	Rolled in scale
5	Scratches

The dataset images are divided into 300 images for each class, 240 for training, and 60 for evaluation. The defect images for each class are as follows.

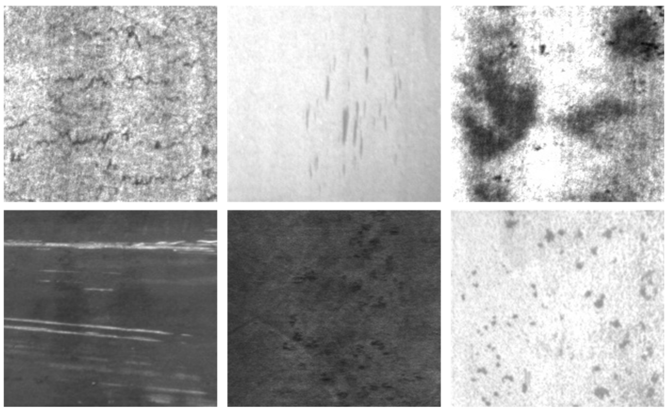


Fig. 2 Defect images (from left-top clockwise, crazing, inclusion, patches, pitted surface, rolled in scale, scratches

The images are all 200×200 in size and are gray images.

C. Designing an Optimal Model

For the basic model design and optimization, we need to simplify the layer representation, so we define conv-activation-pooling as a single layer and set the size of all kernels to 3×3 and the stride to 1×1. To design the model, the framework uses TensorFlow 2.12. The basic model preselects the number of layers in the feature extraction stage and then continuously iterates through training and evaluation, reducing the number of layers until the f1-score reaches a certain level and storing the model. After that, the model is optimized by increasing the number of epochs, and the method is provided in Fig. 3.

Table 2 shows the reduced model structure. In Table 2, an optimization model is designed that uses only four layers in the feature extraction phase and two layers in the classification phase.

TABLE II
OPTIMIZED MODEL STRUCTURE

Layer	Output Size (H, W, C)
Input	(200, 200, 3)
Conv1	(100, 100, 8)
Conv2	(50, 50, 16)
Conv3	(25, 25, 32)
Conv4	(12, 12, 64)
FC1	(1, 16)
FC2	(1, 6)

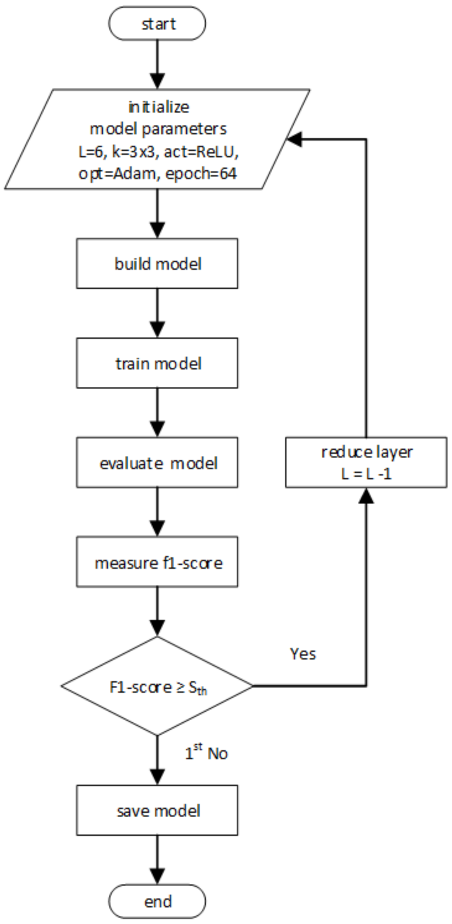


Fig. 3 Model optimization

D. Extra Training

In the previous section, we optimized the model and trained with a reduced viewpoint, so the performance still needs to reach the final goal. Therefore, we use the designed model to improve the performance through additional training. To remove noise, we reduced the learning rate to 0.0001 and increased the number of training epochs to 256 to complete the final model. In Figures 4 and 5, the loss function decreases, and the accuracy gradually increases due to training, indicating that standard training has occurred.

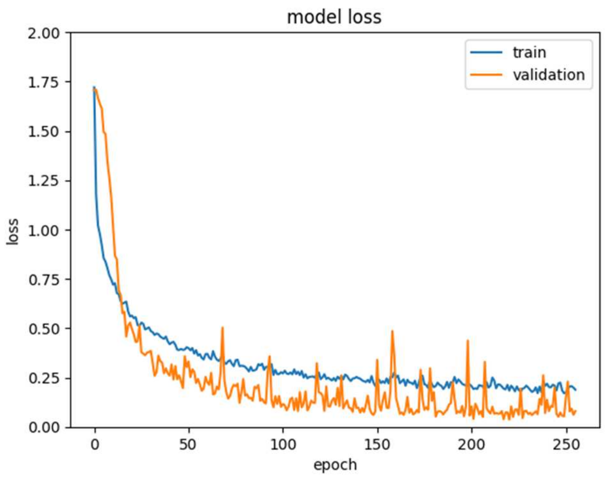


Fig. 4 Model loss while training

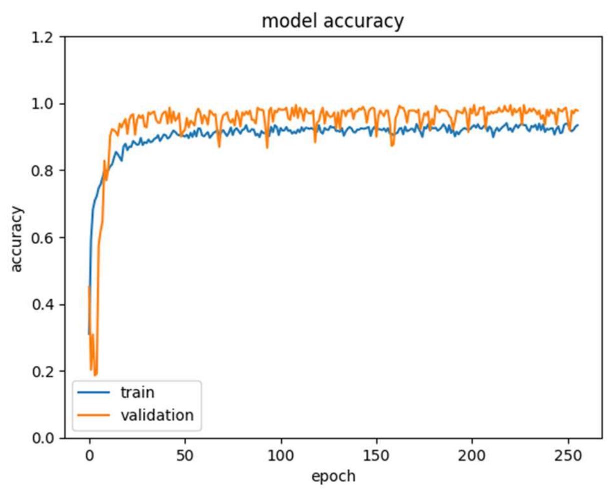


Fig. 5 Model accuracy while training

III. Results and Discussion

Use the evaluation data to test the designed steel surface defect detector. The evaluation data contains 60 images per class, which are applied to the detector to experiment and evaluate the results.

A. Performance Metrics

To evaluate performance in this paper, we use the following performance metrics.

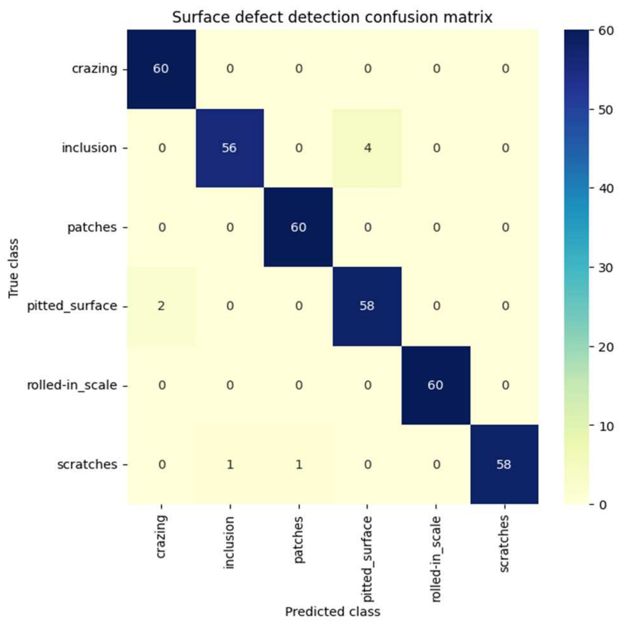
TABLE III
PERFORMANCE METRICS

Name	Definition
TP (true positive)	prediction=positive, correct
FN (false negative)	prediction=negative, false
FP (false positive)	prediction=positive, false
TN (true negative)	prediction=negative, true
precision	TP/(TP+FP)
recall	TP/(TP+FN)
accuracy	TP+TN/(TP+FN+FP+TN)
f1-score	2(precision x recall)/(precision + recall)

A critical metric in Table 3 is the f1-score. This harmonic means of precision and recall is considered the fairest metric in machine learning. The classification results with 60 images in each class are shown in Figure 6 as a chaotic matrix.

The vertical axis is the correct label value, and the horizontal axis is the value determined by the designed detector. The actual and determined or inferred values should be the same in an ideal detector. For example, we fed 60 images for crazing, and the detector labeled all of them as crazing. The 100% performance above is for crazing, patches, and rolled-in scale. In the case of inclusion, the detector incorrectly labeled Fig. 4 as a pitted surface. We obtained performance metrics for each class from this confusion matrix, shown in Table 4.

The results show that the rolled-in scale gets all the answers perfectly correct, while the pitted-in scale performs poorly. Another performance metric is the area under the curve (AUC), which covers the receiver operating characteristic (ROC) curve and its graph. Ideally, the AUC should be 1. Figure 7 shows the ROC curve and AUC simultaneously.



	crazing	inclusion	patches	pitted_surface	rolled-in_scale	scratches
crazing	60	0	0	0	0	0
inclusion	0	56	0	4	0	0
patches	0	0	60	0	0	0
pitted_surface	2	0	0	58	0	0
rolled-in_scale	0	0	0	0	60	0
scratches	0	1	1	0	0	58

Fig. 6 Confusion matrix

TABLE IV
OPTIMIZED MODEL STRUCTURE

	Precision	Recall	F1-Score
Crazing	0.968	1.000	0.984
Inclusion	0.982	0.933	0.957
Patches	0.984	1.000	0.992
Pitted surface	0.935	0.967	0.951
Rolled in scale	1.000	1.000	1.000
Scratches	1.000	0.967	0.983
Average	-	-	0.978

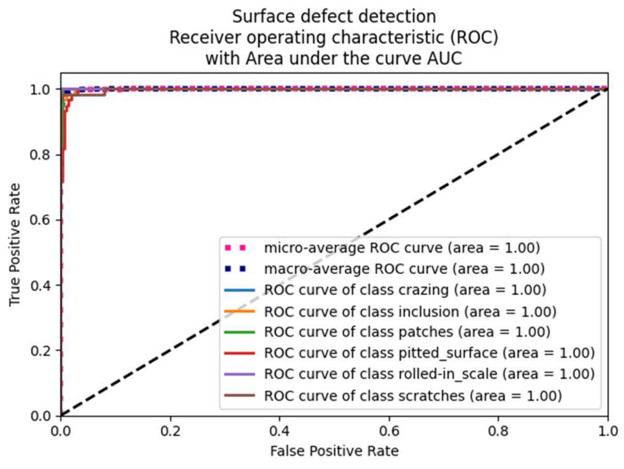


Fig. 7 ROC and AUC

The AUC for all classes is 1.0 because the evaluation tool was designed to round to three decimal places. Therefore, the minimum AUC in this experiment is at least 0.995. The proposed defect detector is designed as an artificial intelligence neural network based on deep learning. The weights of the internal neural network are trained to reason well so that in the final output, the probability in equation (6) is changed to the maximum value by equation (7). Now, we can see how the statistical properties perform when real images are fed in. All images were tested, and Figure 8 shows two test results for each class.

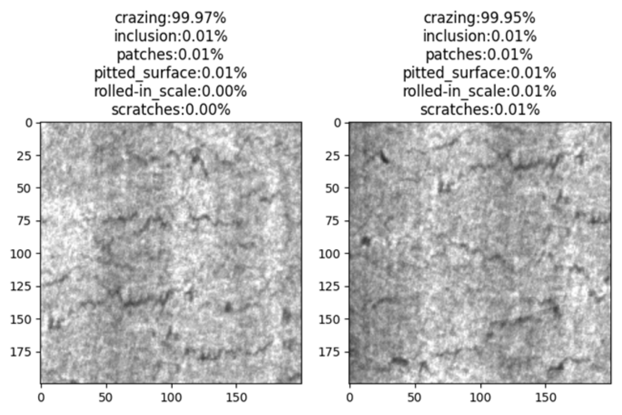


Fig. 8a Real image test crazing

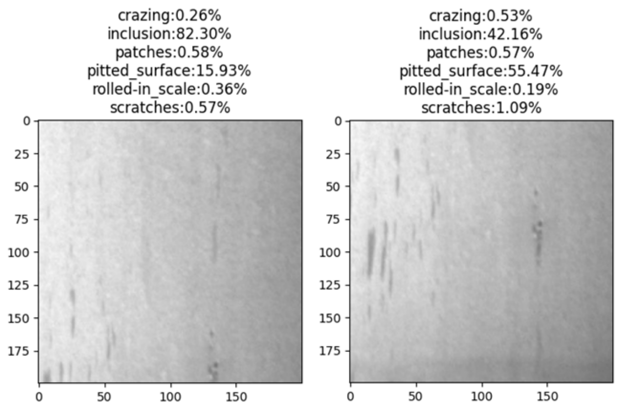


Fig. 8b Real image test inclusion

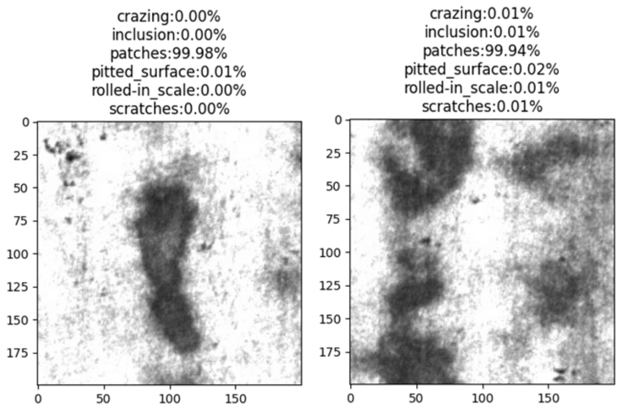


Fig. 8c Real image test patches

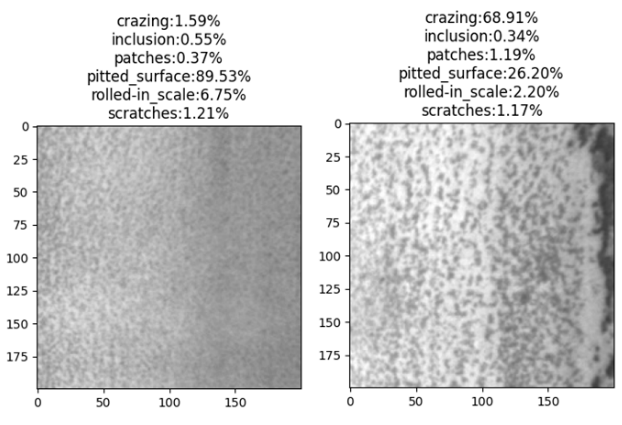


Fig. 8d Real image test pitted surface

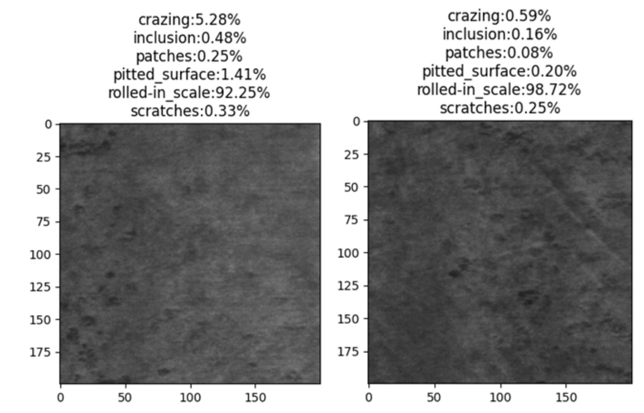


Fig. 8e Real image test rolled in scale

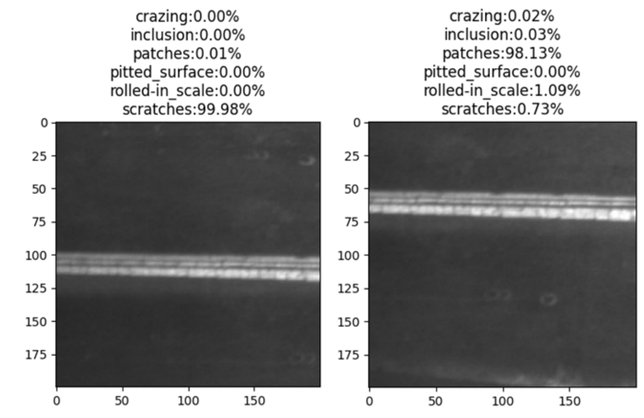


Fig. 8f Real image test scratches

The test image shows all the SoftMax output values, which are the values before the final output, to see what happens probabilistically. Since the number of classes is 6, the sum of the probabilities of each class is either 1 or 100%, and the class with the highest probability value is finally selected. In Figure 8, the images identified as bad are chosen to analyze the causes of inadequate identification. In the inclusion defect experiment, the image on the right with the error is slightly more likely to be a pitted surface defect. Still, the inclusion defect image is similar to the pitted surface defect. Similarly, in the right image in Figure 8.d, the texture is very similar to crazing, even though the label (correct) is a pitted surface. However, the image on the right in Figure 8.f was identified as a patch even though it was a scratch. To analyze the reason for this, we checked all the patch images in the training data set and found a training data set with a similarly straight texture, and we believe that this reduced the error in the inference results. The authors concluded that there is little difference between what a human sees and what an AI sees through this image experiment.

The detector proposed in this paper has shown good performance despite its small size. Table 5 summarizes the comparison with other models using the same dataset.

TABLE V
PERFORMANCE RESULTS

Model	# Layers	# Weight	F1-Score
ResNet152 [26]	152	55M	0.912
ResNet50 [27]	50	55M	0.975
MobileNet-v1 [28]	28	4.2M	< 0.95
our previous [32]	6	0.17M	0.931
proposed	6	0.08M	0.978

The existing detector design with transfer learning requires 4.2M weights and 28 layers, even when using mobileNet with small weights, while the proposed method uses only six layers and 0.08M weights and achieves the best performance. Finally, we measured the inference time. The inference time is measured from the time of reading the image file to the final judgment of the image. Since we do not have a mobile device, we measured the time on a training machine (ubuntu RTX3090) and authorized a total of 60 images × 6 classes = 360 images, and the measured inference time was 1.253772 seconds. This works out to 1.253772/360 = 0.0034827 seconds per image, giving us an inference time of 3.5 msec/image. This leads to the conclusion that if the camera has a field of view (FOV) of 0.5 meters, it can inspect a steel surface moving at a speed of 0.5 / (1.253772/360) ≃ 143.57 m/sec. Of course, in the case of an H beam, which is not a plate and requires multiple views, the time will increase with the number of necessary views.

IV. Conclusion

A cost-competitive and high-performance steel surface defect detector design method was presented. Unlike other papers, we proposed a dedicated model for the actual environment without transfer learning to achieve a small model and fast inference time and showed how to optimize the model. We also provided information on how much process speed should be maintained in the actual field by conducting real image experiments and measuring the inference time, which needed to be presented in previous studies.

Acknowledgment

This work was supported by the Korea Evaluation Institute of Industrial Technology (KEIT) and the Ministry of Trade, Industry & Energy(MOTIE) of the Republic of Korea (No. RS-2022-00155473: Development of energy efficiency improvement and quality improvement technology by applying big data in the section steel rolling process).

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the Innovative Human Resource Development for Local Intellectualization support program (IITP-2024-RS-2023-00260267) supervised by the IITP (Institute for Information & communication Technology Planning & Evaluation).

References

[1] T. Zhang, P. Pan, J. Zhang, and X. Zhang, “Steel Surface Defect Detection Algorithm Based on Improved YOLOv8n,” *Applied Sciences*, vol. 14, no. 12, p. 5325, Jun. 2024, doi:10.3390/app14125325.

[2] I. Konovalenko, P. Maruschak, and V. Brevus, “Steel Surface Defect Detection Using an Ensemble of Deep Residual Neural Networks,” *Journal of Computing and Information Science in Engineering*, vol. 22, no. 1, Jul. 2021, doi: 10.1115/1.4051435.

[3] Z. H. Wei, Y. J. Zhang, X. J. Wang, J. T. Zhou, F. Q. Dou, and Y. H. Xia, "A YOLOV8-based approach for steel plate surface defect detection," *Metalurgija*, vol. 63, no. 1, pp. 28-30, Jan. 2024.

[4] M. Lee and K. Seo, "Comparison of region-based CNN methods for defects detection on metal surface", *Transactions of the Korean Institute of Electrical Engineers*, vol. 67, no. 7, pp. 865-870, 2018.

[5] M. Zhou, W. Lu, J. Xia, and Y. Wang, “Defect Detection in Steel Using a Hybrid Attention Network,” *Sensors*, vol. 23, no. 15, p. 6982, Aug. 2023, doi: 10.3390/s23156982.

[6] C. Lv, Z. Zhang, F. Shen, F. Zhang, and H. Su, “A Fast Surface Defect Detection Method Based on Background Reconstruction,” *International Journal of Precision Engineering and Manufacturing*, vol. 21, no. 3, pp. 363–375, Nov. 2019, doi: 10.1007/s12541-019-00262-2.

[7] B. Feng, J. Wu, H. Tu, J. Tang, and Y. Kang, “A Review of Magnetic Flux Leakage Nondestructive Testing,” *Materials*, vol. 15, no. 20, p. 7362, Oct. 2022, doi: 10.3390/ma15207362.

[8] F. E. Yigzew, H. Kim, S. Mun, and S. Park, “Non-destructive damage detection for steel pipe scaffolds using MFL-based 3D defect visualization,” *Journal of Civil Structural Health Monitoring*, vol. 14, no. 2, pp. 501–509, Nov. 2023, doi: 10.1007/s13349-023-00726-0.

[9] X. Wen, J. Shan, Y. He, and K. Song, “Steel Surface Defect Recognition: A Survey,” *Coatings*, vol. 13, no. 1, p. 17, Dec. 2022, doi:10.3390/coatings13010017.

[10] R. Archana and P. S. E. Jeevaraj, “Deep learning models for digital image processing: a review,” *Artificial Intelligence Review*, vol. 57, no. 1, Jan. 2024, doi: 10.1007/s10462-023-10631-z.

[11] 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, Mar. 2013, doi: 10.1109/tim.2012.2218677.

[12] R. Zaghdoudi, A. Bouguettaya, and A. Boudiaf, “Steel surface defect recognition using classifier combination,” *The International Journal of Advanced Manufacturing Technology*, vol. 132, no. 7–8, pp. 3489–3505, Apr. 2024, doi: 10.1007/s00170-024-13407-z.

[13] K. Demir, M. Ay, M. Cavas, and F. Demir, “Automated steel surface defect detection and classification using a new deep learning-based approach,” *Neural Computing and Applications*, vol. 35, no. 11, pp. 8389–8406, Dec. 2022, doi: 10.1007/s00521-022-08112-5.

[14] K. Liu, N. Luo, A. Li, Y. Tian, H. Sajid, and H. Chen, “A New Self-Reference Image Decomposition Algorithm for Strip Steel Surface Defect Detection,” *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 7, pp. 4732–4741, Jul. 2020, doi:10.1109/tim.2019.2952706.

[15] Y. T. Eugene Chian and J. Tian, “Surface Defect Inspection in Images Using Statistical Patches Fusion and Deeply Learned Features,” *AI*, vol. 2, no. 1, pp. 17–31, Jan. 2021, doi: 10.3390/ai2010002.

[16] Y. S. Gan, S.-S. Chee, Y.-C. Huang, S.-T. Liong, and W.-C. Yau, “Automated leather defect inspection using statistical approach on image intensity,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 10, pp. 9269–9285, Nov. 2020, doi:10.1007/s12652-020-02631-6.

[17] S. Son, and S. Suh, “Deep learning Based Fire and Smoke Detection Systems in Camera,” *Journal of Knowledge Information Technology and Systems*, vol. 17, no. 6, pp. 1251-1258, 2022.

[18] D. Shang, F. Zhang, D. Yuan, L. Hong, H. Zheng, and F. Yang, “Deep Learning-Based Forest Fire Risk Research on Monitoring and Early Warning Algorithms,” *Fire*, vol. 7, no. 4, p. 151, Apr. 2024, doi:10.3390/fire7040151.

[19] P. Dai et al., “Multi-Scale Video Flame Detection for Early Fire Warning Based on Deep Learning,” *Frontiers in Energy Research*, vol. 10, Mar. 2022, doi: 10.3389/fenrg.2022.848754.

[20] H. Kwon, and S. Suh, “Automated Opthalmic Disease Diagnosis System Using Deep Learning,” *Journal of Knowledge Information Technology and Systems*, vol. 17, no. 6, pp. 1089-1097, 2022.

[21] M. Iman, H. R. Arabnia, and K. Rasheed, “A Review of Deep Transfer Learning and Recent Advancements,” *Technologies*, vol. 11, no. 2, p. 40, Mar. 2023, doi: 10.3390/technologies11020040.

[22] V. Chauhan, K. D. Joshi, and B. Surgenor, “Image Classification Using Deep Neural Networks: Transfer Learning and the Handling of Unknown Images,” *Engineering Applications of Neural Networks*, pp. 274–285, 2019, doi: 10.1007/978-3-030-20257-6_23.

[23] K. Glynis, Z. Kapelan, M. Bakker, and R. Taormina, “Leveraging Transfer Learning in LSTM Neural Networks for Data-Efficient Burst Detection in Water Distribution Systems,” *Water Resources Management*, vol. 37, no. 15, pp. 5953–5972, Oct. 2023, doi: 10.1007/s11269-023-03637-3.

[24] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, doi: 10.1109/cvpr.2016.90.

[25] I. Konovalenko, P. Maruschak, V. Brevus, and O. Prentkovskis, “Recognition of Scratches and Abrasions on Metal Surfaces Using a Classifier Based on a Convolutional Neural Network,” *Metals*, vol. 11, no. 4, p. 549, Mar. 2021, doi: 10.3390/met11040549.

[26] S. Wang, X. Xia, L. Ye, and B. Yang, “Automatic Detection and Classification of Steel Surface Defect Using Deep Convolutional

Neural Networks,” *Metals*, vol. 11, no. 3, p. 388, Feb. 2021, doi:10.3390/met11030388.

[27] Y. Li, H. Huang, Q. Xie, L. Yao, and Q. Chen, “Research on a Surface Defect Detection Algorithm Based on MobileNet-SSD,” *Applied Sciences*, vol. 8, no. 9, p. 1678, Sep. 2018, doi: 10.3390/app8091678.

[28] M. Rybczak and K. Kozakiewicz, “Deep Machine Learning of MobileNet, Efficient, and Inception Models,” *Algorithms*, vol. 17, no. 3, p. 96, Feb. 2024, doi: 10.3390/a17030096.

[29] O. I. Aboulola, “Improving traffic accident severity prediction using MobileNet transfer learning model and SHAP XAI technique,” *PLOS ONE*, vol. 19, no. 4, p. e0300640, Apr. 2024, doi:10.1371/journal.pone.0300640.

[30] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications", arXiv:1704.04861, 2017, [online] Available: http://arxiv.org/abs/1704.04861.

[31] S. Suh, “Deep learning Based Surface Defect Detector on Metal Surface,” *Journal of Knowledge Information Technology and Systems*, vol. 16, no. 5, pp. 897-904, 2021.

[32] Z. Li, H. Li, and L. Meng, “Model Compression for Deep Neural Networks: A Survey,” *Computers*, vol. 12, no. 3, p. 60, Mar. 2023, doi:10.3390/computers12030060.

[33] K. Dixit, “NEU Surface Defect Database,” Northeastern University, 2023, [online] Available: https://www.kaggle.com/datasets/kaustubhdikshit/neu-surface-defect-database/