2021 Twelfth International Conference on Ubiquitous and Future Networks (ICUFN) | 978-1-7281-6476-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICUFN49451.2021.9528702

Metal Defect Classification Using Deep Learning

Aji Teguh Prihatno, Ida Bagus Krishna Yoga Utama, Jun Yong Kim*, and Yeong Min Jang, *Member*, IEEE
Department of Electronics Engineering, Kookmin University, Seoul 02707, Korea
*ENS. Co. Ltd, Ansan 15655, Korea
Email: aji.teguh@gmail.com; krishna@kookmin.ac.kr; yong531@ens-km.co.kr;
yjang@kookmin.ac.kr

***Abstract*— In the era of Industry 4.0, the vast development of Smart Factory is always followed by the advancement of Deep Learning technology. To avoid the smart factory system from unwanted losses because of defects in its output production in the steel factory, defect classification on steel sheets based on Deep Learning should be developed precisely. This paper explains how the Deep Learning technique was used to implement defect detection in a smart factory. For this study, we used an open dataset of steel defects. The result of the Deep Learning method for the defect detection system generates 96% accuracy, 0.95 recall, and a precision of 0.97 on the training process. This research goal may contribute to enhancing efficiency and cost reduction in the smart steel factory environment.**

***Keywords*— Smart steel factory, Deep Learning, OpenCV, Kaggle dataset, steel defect classification**

I. INTRODUCTION

Steel sheets in the production process may contain defects such as scratches, scars, black burn, bug prints, inclusions, bright prints, burrs, iron scales, seams, etc., due to the influence of raw materials, rolling process, and system control, among other factors [1]. By detecting, localizing, recognizing, and then rectifying causal variables, the defect detection system in steel sheet surfaces plays a crucial part in the steel sheet business. The detection procedure entails determining the presence of steel surface flaws using images captured by industrial cameras [2].

However, the inspection of the quality of steel sheets is usually done manually. Manual inspection of steel defects may cause delays in the manufacturing process, and it is not an efficient technique to ensure defect-free steel production. The inspectors' experience may influence the accuracy with which they identify the steel problem.

Manual inspection of steel defects is a time-consuming and inconvenient method in which examiners must hand-inspect each steel product. Steel sheets defect detection by inspectors is low on a unit per hour (UPH) basis. As a consequence of the delay, production rates will be declined, impacting significant economic losses for industrial enterprises [3]. Automatic steel defect identification is critical for steel manufacturing organizations as an important part of Industry 4.0 which integrate IoT, big data, and artificial intelligence (AI) [4]. Machine vision-based solutions were being developed to overcome the inefficiencies of manual inspection aforementioned. The defect pattern from steel sheets can be seen in the manufacturing steel industry by Deep Learning methods.

These methods were used to improve results when dealing with large amounts of steel image data. To address this issue, many researchers have studied machine learning techniques for steel defect detection. There are many methods that have been proposed to train neural network models in machine vision applications for defect detection, such as YOLO(you only look once), SSD(single shot detectors), and CNN(convolutional neural network) [5].

In this study, we implement Deep Learning using Convolutional Neural Network to detect the defect on steel sheets in the steel factory. This research goal is to enhance maintenance and lowering the cost in the steel factory, while undetected defects in the steel sheets can decrease the quality of output production.

This paper comprises six parts. Chapter II explains the dataset and features. Chapter III talks about the methodology which contains the convolutional neural network method as a basis for defect detection, while chapter IV shares about result and analysis after training, validation, and testing dataset, and chapter V share the conclusion.

II. DATASET AND FEATURE

Steel sheet images, related defect types, and defect locations make up our dataset [6]. Images of steel sheets are divided into three categories: training, validation, and testing. There are 9390 for training images, 1657 for validation images, and 1950 for testing images.

From all 12997 datasets, there are 7095 images have defects, whereas 5902 images are defect-free. Figure 1 shows the distribution of all training photos according to the four defects which class 3 has the highest number of defect images, 56% of the total defect images, and class 1 has the lowest number of defect images, 0.03% of the total defect images. The defective sample images of each type from our dataset are presented in Fig. 2.

Authorized licensed use limited to: Pusan National University Library. Downloaded on August 01,2022 at 05:23:51 UTC from IEEE Xplore. Restrictions apply.

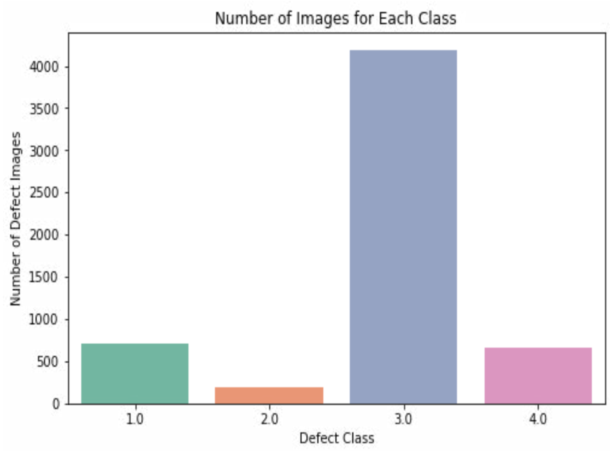

Figure 1. Number of images for each class.

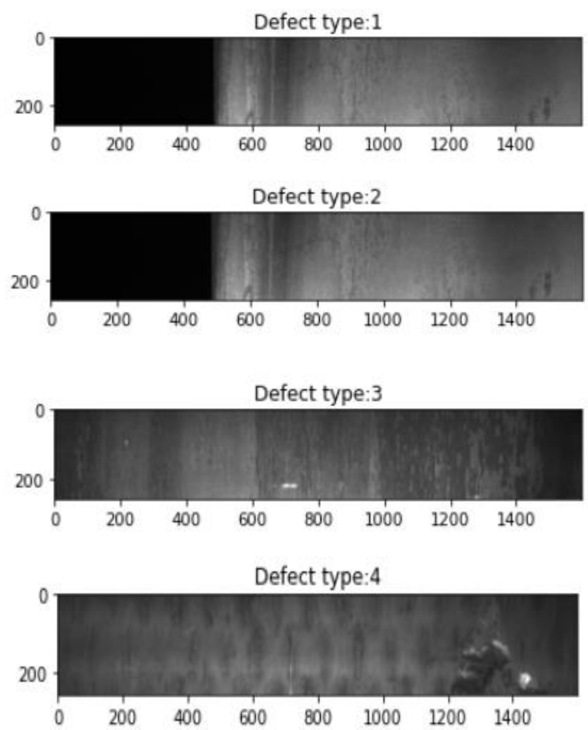

Figure 2. Sample images of each defect type.

III. METHODOLOGY

Convolutional neural networks (CNN) use a convolutional function in one or more of their layers. The feature extractor subsystem and the classifier subsystem make up the architecture of a convolutional neural network. The most significant aspect that distinguishes a convolutional neural network from another feature extractor method is the feature extractor subsystem. The convolutional layer is the major component that extracts all of the information from the tensor input data, while the pooling layer reduces the data's spatial size. The output of the feature extractor subsystem is sent into the classifier subsystem, which is the second half of the convolutional neural network [7].

The convolutional layer, which is part of the feature extractor subsystem, extracts the feature from the tensor input data. The tensor input data undergoes a convolution process in the convolutional layer. The forward pass equation is given by the following formula:

$$F_1[c_0] = \sum_{ci=1}^{m} F_0[c_i] \circledast W_1[c_0, c_i] \tag{1}$$

The F_0 is the output from the previous layer with c_i = 1,2,3, …, m that denotes the input channel and $\circledast$ denotes a 2-dimensional convolution operation. Meanwhile, the F_1 is feature map output which c_0 = 1,2,3, …, n that denotes the output channel. The W_1 is shared the weight of the convolutional layer on c_0 channel.

In the backward pass on convolution layer, there are two updates achieved, i.e., deltas and weight connection, that demonstrated by

$$\delta F_0[c_i] = \sum_{co=1}^{n} \delta\, F_1[c_o] \circledast f_{rot}180(W_1)[c_o, c_i] \tag{2}$$

$$\delta W_1[c_o, c_i] = F_0[c_i] \circledast \delta F_1[c_o] \tag{3}$$

The updated delta denoted by $\delta F_0[c_i]$ with $c_i = 1,2,3, \ldots, m$ where m is the number of the used channel. $\delta F_1[c_o]$ is the delta from the previous layer with $c_o = 1,2,3, \ldots, n$ where n is the number of its channels [7].

The CNN architectures are used to categorize steel sheet defect images into four different classes in this paper. The CNN architecture is commonly divided into two stages: feature extraction and classification, with the latter being learnt to describe spatial information of pictures across layers. Extracted feature representations are supplied into the architecture's final stage, where the model calculates the likelihood of belonging to a specific class [8].

In the computation, the authors used 4 convolutional layers and 6 dense layers with various numbers of neurons. To extract information during deep network training, the convolutional neural network architecture includes a convolutional layer to extract the information on the data. The convolutional layer allows the network to gain better and more accurate information from the data and can improve the model's performance. The convolutional neural network used in this work is consist of 4 convolutional layer which each convolutional layer consists of convolution, activation by using ReLU, and pooling using maximum pooling method. Meanwhile, for the dense layer, this work uses 6 dense layers with various numbers of neurons on each layer.

Authorized licensed use limited to: Pusan National University Library. Downloaded on August 01,2022 at 05:23:51 UTC from IEEE Xplore. Restrictions apply.

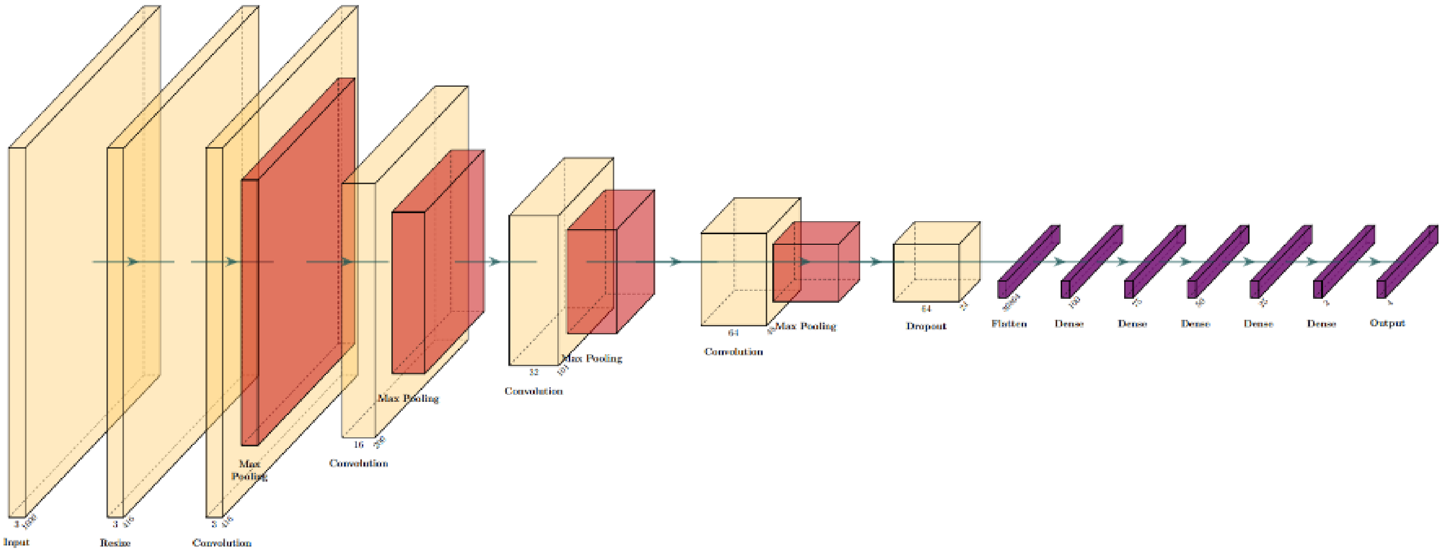

Figure 3. Convolutional neural network (CNN) architecture for metal defect classification

TABLE I. CNN BASEMODEL.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 416, 416, 3)	84
activation (Activation)	(None, 416, 416, 3)	0
max_pooling2d (MaxPooling2D)	(None, 208, 208, 3)	0
conv2d_1 (Conv2D)	(None, 206, 206, 16)	448
activation_1 (Activation)	(None, 206, 206, 16)	0
max_pooling2d_1 (MaxPooling2	(None, 103, 103, 16)	0
conv2d_2 (Conv2D)	(None, 101, 101, 32)	4640
activation_2 (Activation)	(None, 101, 101, 32)	0
max_pooling2d_2 (MaxPooling2	(None, 50, 50, 32)	0
conv2d_3 (Conv2D)	(None, 48, 48, 64)	18496
activation_3 (Activation)	(None, 48, 48, 64)	0
max_pooling2d_3 (MaxPooling2	(None, 24, 24, 64)	0
dropout (Dropout)	(None, 24, 24, 64)	0
flatten (Flatten)	(None, 36864)	0
dense (Dense)	(None, 100)	3686500
dense_1 (Dense)	(None, 75)	7575
dense_2 (Dense)	(None, 50)	3800
dense_3 (Dense)	(None, 25)	1275
dense_4 (Dense)	(None, 2)	52
dense_5 (Dense)	(None, 4)	12

Total params: 3,722,882
Trainable params: 3,722,882
Non-trainable params: 0

The neuron number is 100, 75, 50, 25, 2, and 4, respectively for each dense layer from first dense layer to output layer. The output layer is consisted of 4 neurons to represent 4 classes of defects that we want to classify.

Figure 3 shows CNN architecture for metal defect classification which elaborated by Table I that shows the basemodel of CNN which indicates the number of parameters.

IV. RESULT AND ANALYSIS

To read the image and convert it to RGB (red-green-blue) format, we use OpenCV (open source computer vision). OpenCV is an open-source library for image and video analysis, originally introduced by Intel [9].

OpenCV comes with a number of tools for dealing with computer vision issues. It includes low-level image processing functions as well as high-level face detection, feature matching, and tracking algorithms. The following are some of the most common image processing techniques: feature detection, image filtering, image transformation, and object tracking [10].

Feature Detection:

Many computer vision algorithms employ features as a starting point. A feature is defined as a "interesting" element of a picture. Because features serve as the starting point and primary primitives for future algorithms, the total algorithm will frequently be judged only on the quality of its feature detector. The process of discovering certain features of a visual stimuli is known as feature detection.

Image Filtering:

It is a method of altering or improving an image. There are two forms of image filtering. The first is linear image filtering, in which an output pixel's value is a linear combination of the pixels in the input pixel's neighborhood. The second type is non-linear image filtering, in which the output value is not a linear function of the input value.

Authorized licensed use limited to: Pusan National University Library. Downloaded on August 01,2022 at 05:23:51 UTC from IEEE Xplore. Restrictions apply.

Image Transformation:

Image transformation creates a "new" image from two or more sources that is superior to the original input images in terms of highlighting specific aspects or properties of interest. Simple arithmetic operations are applied to the image data in basic image transformations. Image subtraction is frequently used to detect differences between photos gathered on different dates.

Main image transformation methods are The Hough Transform is used to find lines in an image, while the Radon Transform is used to reconstruct images from fan-beam and parallel-beam projection data. The Discrete Cosine Transform is used in image and video compression, while the Discrete Fourier Transform is used in filtering and frequency analysis. The last transformation is Wavelet Transform that used to perform discrete wavelet analysis, fuse images, and denoise [10].

Object Tracking:

The practice of locating an object (or numerous objects) throughout a few frames captured in a fixed interval of time is known as object tracking [11]. It is a critical component in a wide range of computer vision applications, including surveillance, human-computer interaction, and medical imaging.

The following are the OpenCV modules for image processing applications:

IMGPROC module includes image processing functions like linear and non-linear image filtering, as well as geometrical picture modifications.

ML module consist of machine-learning interfaces.

CORE includes the fundamental data structures and functions that other modules rely on.

HighGUI module provides basic I/O interfaces as well as multiplatform windowing.

VIDEO module contains algorithms for motion estimation and object tracking.

By measuring the classification report, we can get loss, accuracy, precision, and recall. The ratio of recognized actual flaws to all detected defects is called precision p. The ratio of discovered true defects to all true defects is called recall r, and accuracy (ACC). The terms p, r, and ACC are defined as follows:

$$p = \frac{TP}{TP + FP} \tag{4}$$

$$r = \frac{TP}{TP+FN} \tag{5}$$

$$ACC = \frac{TP+TN}{TP+FP+FN+TN} \tag{6}$$

Adam optimizer was used in our model because it can achieve the shortest convergence time [12]. We also use the focal Tversky loss function during training to get the optimum precision and recall balance in semantic segmentation [13]. After epoch 99, the CNN model indicates good achievement for predicting defects of all classes as shown in Figure 4.

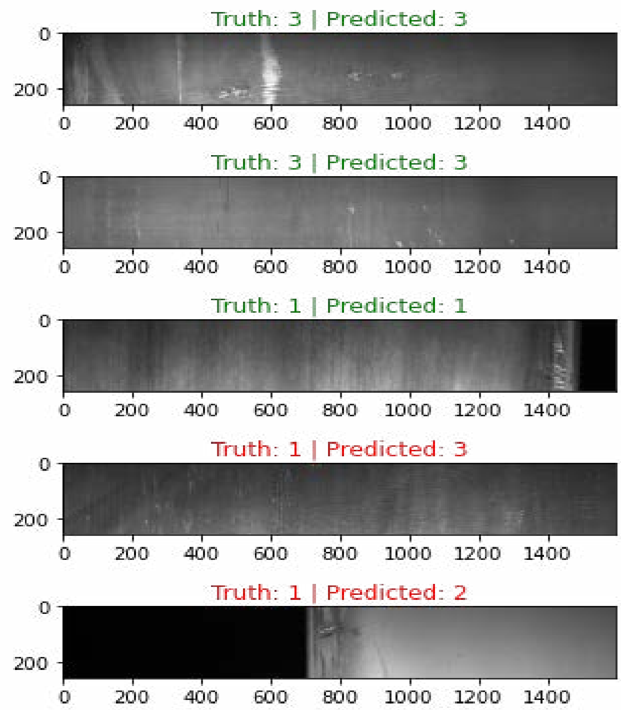

Figure 4. Classification result.

Table II shows the training and testing result for steel sheet defect classification.

TABLE II. CLASSIFICATION REPORT WITH PRECISION

	Loss	Accuracy	Precision	Recall
Training	0.0415	0.96	0.97	0.95
Testing	0.9	0.73	0.74	0.7

V. CONCLUSION

This paper presents the implementation of steel sheet defect detection by using a convolutional neural network and the OpenCV method. To identify steel surface defects and perform region-based multiple label classification, the proposed work proves that the convolutional neural network is able to perform classification for the steel sheet defect with good results. The convolutional neural network achieves 96% accuracy in the training process and 73% accuracy in the testing process. From the result achieved, this study lays the groundwork for in-situ advanced manufacturing, real-time online monitoring, and defect-controlling operations. In the future, we will study additional deep neural networks on this topic in the future, examine their performance, and enhance classification accuracy.

ACKNOWLEDGMENT

This work was partly supported by the Technology development Program of MSS [S3098815] and the MSIT

Authorized licensed use limited to: Pusan National University Library. Downloaded on August 01,2022 at 05:23:51 UTC from IEEE Xplore. Restrictions apply.

(Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2021-0-01396) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation).

REFERENCES

[1] J. Li, Z. Su, J. Geng, and Y. Yin, “Real-time Detection of Steel Strip Surface Defects Based on Improved YOLO Detection Network,” *IFAC-PapersOnLine*, vol. 51, no. 21, pp. 76–81, 2018, doi: 10.1016/j.ifacol.2018.09.412.

[2] D. Amin and S. Akhter, “Deep Learning-Based Defect Detection System in Steel Sheet Surfaces,” *2020 IEEE Region 10 Symposium, TENSYMP 2020*, no. June, pp. 444–448, 2020, doi: 10.1109/TENSYMP50017.2020.9230863.

[3] M. Abu, A. Amir, Y. H. Lean, N. A. H. Zahri, and S. A. Azemi, “The Performance Analysis of Transfer Learning for Steel Defect Detection by Using Deep Learning,” *Journal of Physics: Conference Series*, vol. 1755, no. 1, 2021, doi: 10.1088/1742-6596/1755/1/012041.

[4] A. T. Prihatno, H. Nurcahyanto, and Y. M. Jang, “Smart Factory Based on IoT Platform,” *KIC Summer Conference 2020*, pp. 2–4, 2020, doi: 10.3390/MACHINES6020023.Thalesgroup.

[5] A. A. Tulbure, A. A. Tulbure, and E. H. Dulf, “A review on modern defect detection models using DCNNs – Deep convolutional neural networks,” *Journal of Advanced Research*, no. xxxx, 2021, doi: 10.1016/j.jare.2021.03.015.

[6] “Severstal Steel Defect Detection,” 2019. https://www.kaggle.com/c/severstal-steel-defect-detection/data (accessed Jul. 10, 2021).

[7] I. B. K. Y. Utama, A. Faqih, and B. Kusumoputro, “Three mixture of odor classification using convolutional neural network,” *2019 16th International Conference on Quality in Research, QIR 2019 - International Symposium on Electrical and Computer Engineering*, pp. 5–8, 2019, doi: 10.1109/QIR.2019.8898255.

[8] S. Y. Lee, B. A. Tama, S. J. Moon, and S. Lee, “Steel surface defect diagnostics using deep convolutional neural network and class activation map,” *Applied Sciences (Switzerland)*, vol. 9, no. 24, 2019, doi: 10.3390/app9245449.

[9] I. Culjak, D. Abram, T. Pribanic, H. Dzapo, and M. Cifrek, “A brief introduction to OpenCV,” *MIPRO 2012 - 35th International Convention on Information and Communication Technology, Electronics and Microelectronics - Proceedings*, pp. 1725–1730, 2012.

[10] M. Naveenkumar and V. Ayyasamy, “OpenCV for Computer Vision Applications,” *Proceedings of National Conference on Big Data and Cloud Computing (NCBDC’15)*, no. March 2015, pp. 52–56, 2016, [Online]. Available: https://www.researchgate.net/publication/301590571_OpenCV_for_Computer_Vision_Applications.

[11] G. Chandan, A. Jain, H. Jain, and Mohana, “Real Time Object Detection and Tracking Using Deep Learning and OpenCV,” *Proceedings of the International Conference on Inventive Research in Computing Applications, ICIRCA 2018*, no. Icirca, pp. 1305–1308, 2018, doi: 10.1109/ICIRCA.2018.8597266.

[12] I. Kandel, M. Castelli, and A. Popovič, “Comparative Study of First Order Optimizers for Image Classification Using Convolutional Neural Networks on Histopathology Images,” *Journal of Imaging*, vol. 6, no. 9, 2020, doi: 10.3390/JIMAGING6090092.

[13] N. Abraham and N. M. Khan, “A novel focal tversky loss function with improved attention u-net for lesion segmentation,” *Proceedings - International Symposium on Biomedical Imaging*, vol. 2019-April, pp. 683–687, 2019, doi: 10.1109/ISBI.2019.8759329.

Authorized licensed use limited to: Pusan National University Library. Downloaded on August 01,2022 at 05:23:51 UTC from IEEE Xplore. Restrictions apply.