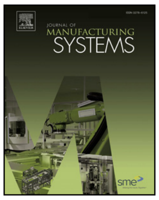
Journal of Manufacturing Systems 55 (2020) 317–324

Contents lists available at ScienceDirect

Journal of Manufacturing Systems

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



Automated defect inspection system for metal surfaces based on deep learning and data augmentation

Jong Pil Yun[a,*], Woosang Crino Shin[a], Gyogwon Koo[b], Min Su Kim[c], Chungki Lee[d], Sang Jun Lee[c]

[a] *Korea Institute of Industrial Technology, Daegyeong Division, Daegu, Republic of Korea*
[b] *Daegu-Gyeongbuk Institute of Science and Technology (DGIST), Daegu, Republic of Korea*
[c] *Pohang University of Science and Technology, Department of Electrical Engineering, Pohang, Republic of Korea*
[d] *Yujin Instec Core Co. Ltd., Seoul, Republic of Korea*

ARTICLE INFO

Keywords:
Defect detection
Vision inspection
Machine learning
Metal surfaces
Deep learning

ABSTRACT

Recent efforts to create a smart factory have inspired research that analyzes process data collected from Internet of Things (IOT) sensors, to predict product quality in real time. This requires an automatic defect inspection system that quantifies product quality data by detecting and classifying defects in real time. In this study, we propose a vision-based defect inspection system to inspect metal surface defects. In recent years, deep convolutional neural networks (DCNNs) have been used in many manufacturing industries and have demonstrated the excellent performance as a defect classification method. A sufficient amount of training data must be acquired, to ensure high performance using a DCNN. However, owing to the nature of the metal manufacturing industry, it is difficult to obtain enough data because some defects occur rarely. Owing to this imbalanced data problem, the generalization performance of the DCNN-based classification algorithm is lowered. In this study, we propose a new convolutional variational autoencoder (CVAE) and deep CNN-based defect classification algorithm to solve this problem. The CVAE-based data generation technology generates sufficient defect data to train the classification model. A conditional CVAE (CCVAE) is proposed to generate images for each defect type in a single CVAE model. We also propose a classifier based on a DCNN with high generalization performance using data generated from the CCVAE. In order to verify the performance of the proposed method, we performed experiments using defect images obtained from an actual metal production line. The results showed that the proposed method exhibited an excellent performance.

1. Introduction

Recently, interest in developing smart factories within the manufacturing industry has grown rapidly. In general, a smart factory can be defined as a factory that operates base on advanced information and communications technologies (ICT), in which all the elements in the factory are connected organically and operated intelligently. In order to optimize production conditions and to develop the technology needed to create products using minimal cost and time, it is important to measure the product quality in real time and store the measurement in a data-base. For this reason, a defect inspection system that detects defects in real time and classifies defect types is one of the key technologies required for implementing the smart factory. This technology is particularly important for the following reasons. First, by classifying the defects correctly, it is possible to analyze the type of defect occurrence according to the process conditions, thereby optimizing the process conditions. Second, equipment failure can be diagnosed by analyzing the type of defect occurrence. For example, in the case of rolling, when a defect occurs in a specific cycle, the particular roll that caused the defect can be identified by analyzing the cycle. Third, it is possible to determine the grade of the product by analyzing the number and type of defect occurrences, so that it can be decided whether or not to ship the product, based on whether or not it meets the requirements of the customer. Finally, it can prevent sending out bad products to customers, minimizing customer claims, and maximizing company reliability.

However, despite this importance, many manufacturers still rely on visual and sampling inspection by an operator, as opposed to total inspection in real-time. Sampling inspection is not suitable for process optimization because the operational data and product quality data are not matched one to one. Unlike an off-line sampling test, total inspection in real-time can prevent mass defects in advance. In addition, the

* Corresponding author.
E-mail address: rebirth@kitech.re.kr (J.P. Yun).

https://doi.org/10.1016/j.jmsy.2020.03.009
Received 20 May 2019; Received in revised form 24 March 2020; Accepted 24 March 2020
Available online 15 May 2020

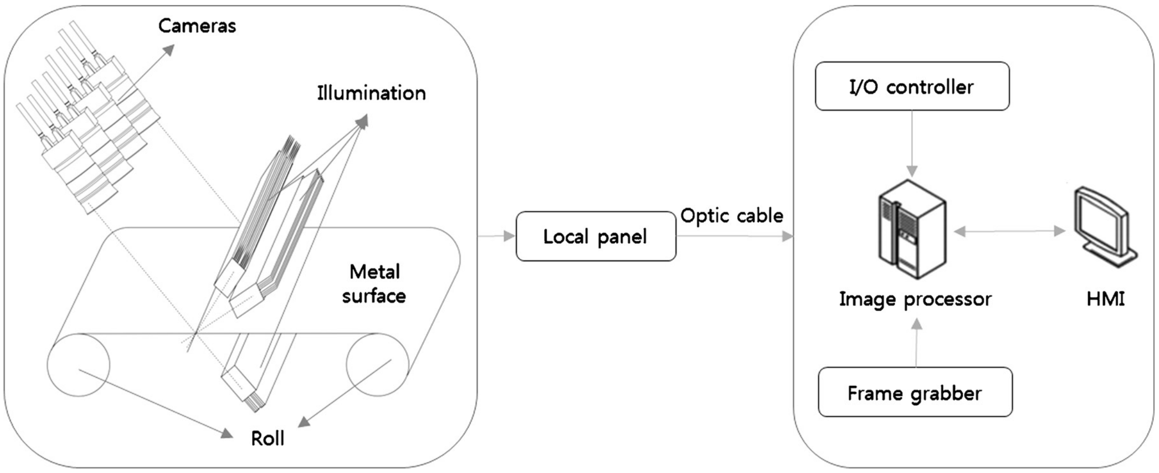


Fig. 1. System configuration of the proposed defect inspection for metal surfaces.

automatic defect inspection system has many advantages compared to a human inspector, such as continuous operation over a long period of time, consistent inspection results, and operation in a harsh environment involving high temperature and dust. For this reason, vision-based systems that automatically detect defects in products have been studied in various industries such as image sensors [1], bottles [2], IC chips [3], fabrics [4]-[5], LCDs [6], car body surfaces [7], magnetic tiles [8], concrete bridges [9], fruits [10], ceramic tiles [11], machines [12]-[13], metals [14] and railways [15].

The object of this paper is the defect of metal products. In the metal industry, it is also important to inspect the defects and feed back to the work with information such as the name, size, shape of the classified defects and the location of each defect. This paper focuses on defect inspection of products in the product quality control process. Defect inspection of metal is also being studied continuously by applying various image processing technology and machine vision technology [16]-[24]. A novel algorithm named discriminant manifold regularized local descriptor was proposed to conduct the defect classification for steel surface [16]. Park et al. [17] proposed periodic defect inspection system for steel wire rods based on Haar undecimated discrete wavelet transform and analysis of the frequency spectrum. A surface inspection system for planar steel based on multispectral photometric stereo technique [18]. To detect pinholes of slab, Gabor filter combination was presented [19]. A framework for the defect inspection of billet surface has been proposed to minimize the impact of non-uniform illumination [20].

Recently, defect inspection methods using deep neural networks (DNNs) have also been applied to the metal industries [21]-[24]. A deep learning algorithm based on convolutional neural network (CNN) was proposed to detect three different welding defects based on the captured signals by the multi-sensor system [21]. A novel cascaded autoencoder architecture was proposed for segmenting and localizing defects [22]. Yi et al. [23] proposed CNN to extract defect features and classify defect categories. A simple model of CNN was presented to classify defects of steel sheet [24]. To improve classification performance, the structure of neural networks must be sufficiently deep. Deep neural networks requires a lot of data to train. However, in actual applications, it is often difficult to obtain considerable defect data. Particularly, there is an imbalanced data problem wherein data pertaining to a specific defect is insufficient compared to the data available for other defects. Under these conditions, the classification model becomes an overfitting model and generalization performance is reduced. Generally, a data augmentation method is used to solve the above-mentioned problem. In the case of image classification, data augmentation focuses on generating image data through label-preserving linear transformations, such as translation, rotation, scaling, blur, contrast, horizontal shearing, and so on. However, they only lead to an image-level transformation through depth and scale and are no actually helpful for dividing a clear boundary between data manifolds. Such data augmentation does not improve data distributions which is determined by higher-level features [25,26].

In this study, we propose a defect classification algorithm for metal surfaces using a CNN model. In particular, to solve the imbalanced defect data problem and to improve classification performance, we propose a data augmentation method using the conditional convolutional variational autoencoder (CCVAE) in place of a general data augmentation method such as rotation, translation, or noise addition. After learning the distribution of a given defect data using CCVAE, various defect images are generated by sampling within the learned distribution. We attempt to prove that the accuracy of classification is improved by training the classification model with deep neural networks using the generated defect image data. To demonstrate the effectiveness of the proposed method, actual metal surface images were acquired by an optical system and experiments were performed on the obtained images. In particular, we set the amount of training data to various conditions and analyzed the classification performance of the model according to the amount of data.

The remainder of this paper is organized as follows. In Section 2, an overview of the overall system for detecting metal defects is introduced. Moreover, the obtained metal surface defects are analyzed. Section 3 deals with data augmentation for classification by CCVAE and the defect classification algorithm using deep convolution neural networks. Using the actual metal surface defect data, we prove the effect of the proposed approach in Section 4. Finally, the conclusion is presented in Section 5.

2. System configuration

The overall configuration of the vision-based inspection system for defects on metal surfaces is shown in Fig. 1. To inspect the surface of moving metal products, a line scan camera suitable for moving objects is used. In this study, the line scan camera used has a resolution of 12288 pixels in the width direction of the object. The collimated illumination is used to increase the signal to noise ratio of the surface defects. Because the line scan camera should operate according to the speed of the moving object, it synchronizes with the speed information of the object. The image obtained from the line scan camera is transmitted to the image processing PC through the frame grabber. The image processing PC analyzes real-time incoming images, detects and classifies defects, and stores the results in a database (DB). Furthermore, the result is displayed on the human-machine interface (HMI) screen, and defect information is provided to the operator.

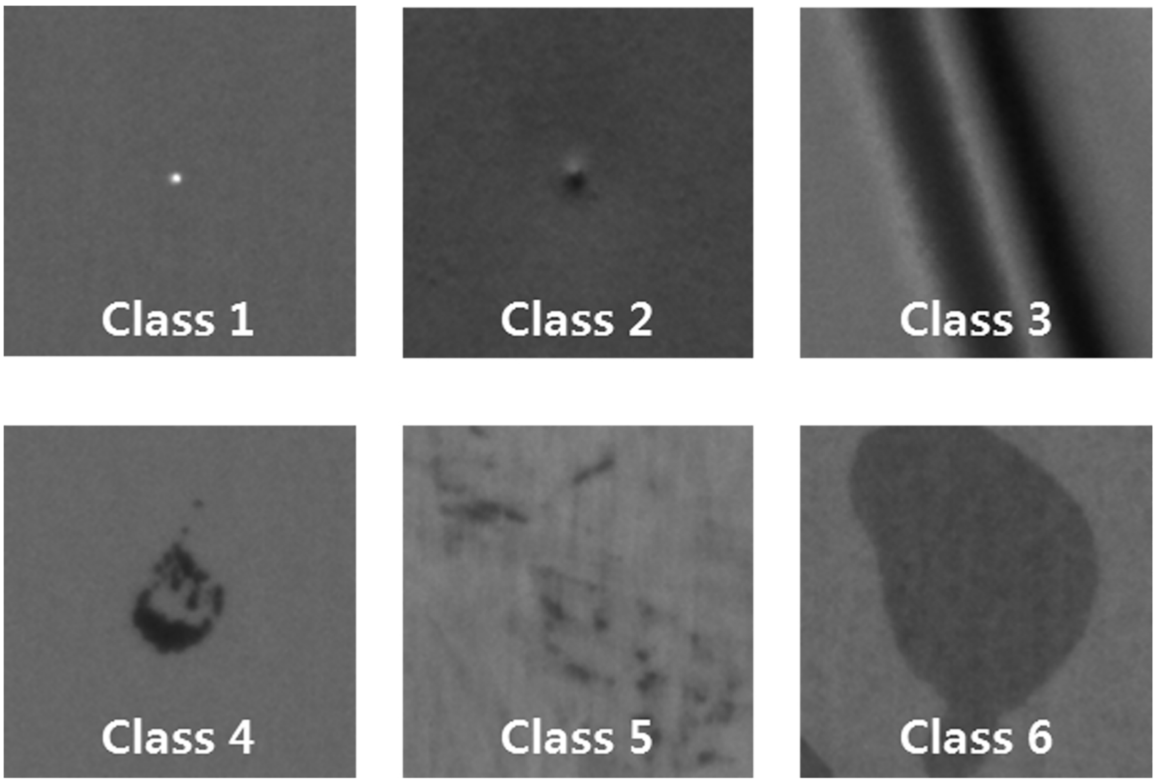


Fig. 2. Metal surface images with defects.

Table 1
Number of images for each class of defects obtained in the proposed defect inspection system.

	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Sum
Test	151	152	126	88	32	106	655
Training	151	152	127	88	32	107	657

Fig. 2 shows the defect images for the metal surfaces that were obtained from the proposed system. The size of the obtained defect images is 120 × 120 (width × height) pixels and consists of six types of defects, as shown in Fig. 2. The brightness of the background also varies. The status of the defect images obtained is shown in Table 1. There are 657 samples used for training and 655 samples for testing. As shown in the table, the number of images collected for each defect is different, and the amount of Class 5 data is significantly smaller than that of Classes 1 and 2. We propose a new defect classification algorithm using CCVAE and DCNN in order to solve the problem of imbalanced data between classes and to improve defect classification performance. A detailed description of the algorithm is given in the below section.

3. Defect classification algorithm

The proposed defect classification algorithm consists of a data generation module and a defect classification module. The image data obtained by the machine vision system is labeled for each defect. The VAE-based data generation model is trained using the defect images and label data. After generating enough data to train a deep learning-based classification model, the classification model is trained by using the generated data. The classification model is trained to minimize the difference between the label information of the defect data and the classification model output value. The trained classification model is installed in the inspection system, to classify the defect images of metal surfaces in real-time.

When developing an automatic defect classification algorithm based on a data-driven approach, it is important to have the same and sufficient amount of data for each defect. However, if an imbalance problem occurs in which the amount of data is different for each class, the generalization performance of the algorithm is degraded. When training a deep model using a small amount of data, the model only works well for given data. In an actual test, overfitting occurs and the performance drops sharply whenever a slight change in data occurs. Among the various ways to solve the overfitting problem, a method to obtain training data by data augmentation is used. In general, the augmentation method for data classification uses flipping, rotation, shifting, addition of noise, blurring, sharpening, and so on. However, there are restrictions when using this method in actual manufacturing. For example, in the rolling process, defects are generated in the longitudinal direction, which is the moving direction of the object. However, if data augmentation of these defects by the rotation method is used, different types of defects will be generated. Eventually, training with such data degrades defect classification performance. In other words, the above data augmentation has a limitation that must be considered after analyzing the causes and types of defects in each manufacturing industry. In this study, augmentation of images is performed by a variational autoencoder method which can generate images by learning the distribution of the given data itself. Based on the generated data, we propose a CNN based classification model with high generalization performance. A detailed description is given in the next section.

3.1. CCVAE

The variational autoencoder (VAE) is a directed graphical model with certain types of latent variables, such as Gaussian latent variables [27], [28]. The VAE can be divided into three parts as shown in Fig. 3: encoder, decoder, and sampling part. The VAE consists of an encoder that extracts features from input x and expresses latent variable z, which is the compressed information of the input. It also contains a

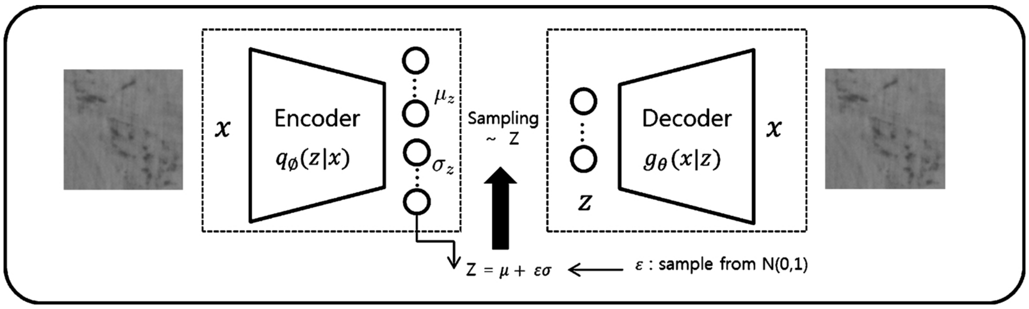


Fig. 3. VAE architecture.

decoder, which generates an image from the latent variable z. The sampling part generates the latent variable z by sampling the output of the encoder defined by μ and σ of the Gaussian distribution. The encoder and decoder of the VAE can be composed of deep neural networks. The loss function for learning these two networks is shown below [29].

$$L_i(\phi, \theta, x_i) = -E_{q_\phi(z|x_i)}[\log(p(x_i|g_\theta(z)))] + \mathrm{KL}(q_\phi(z|x_i)||p(z)) \tag{1}$$

where $p(x|g_\theta(z)) = p_\theta(x|z)$ is the the likelihood of the data x given the latent variable z. $q_\phi(z|x)$ is the approximate posterior and $p(z)$ is the prior distribution of the latent variable z. To make sampling easy, the posterior distribution is usually parameterized by a Gaussian distribution with its mean and variance predicted by the encoder [30]. The first term of Eq. (1) can be understood in terms of the reconstruction of x through the posterior distribution $q_\phi(z|x)$ and the likelihood $p(x|g_\theta(z))$. The second term of Eq. (1) is the Kullback–Leibler (KL) divergence between the approximate posterior and the prior of the latent variable z. This term forces the posterior distribution to be similar to the prior distribution, working as a regularization term [31]. The prior is typically chosen to be a Gaussian with zero mean and unit variance [30].

In this study, we propose a CCVAE structure to generate image data for each defect. The CCVAE uses the same convolutional VAE (CVAE) structure based on an encoder and decoder, but here is a difference, in that it uses information from each class as conditions. The difference is that to generate image data for each type of defect, a defect label composed of one-hot encoding is concatenated with latent variable z to be input to the decoder.

Figs. 4 and 5 show the structure of the data generation model based on the CCVAE, which is proposed in this study, to solve the defect data imbalance. CCVAE consists of an encoder, sampling part, and decoder. The details of the CCVAE architecture are summarized in Tables 2 and 3 . Table 2 shows the structure of the encoder, and the structure of the decoder is summarized in Table 3. The encoder consists of one basic block consisting of a convolution layer, which has 3×3 filters, a batch normalization layer, a rectified linear unit (ReLU) activation function, and a 2×2 max pooling layer, as shown in Fig. 4. The stride of the convolution is set to 1. The stride of max pooling is set to 2. The ReLU function is defined as

$$f(z) = \begin{cases} 0 & \text{, if } z \leq 0 \\ z & \text{otherwise} \end{cases} \tag{2}$$

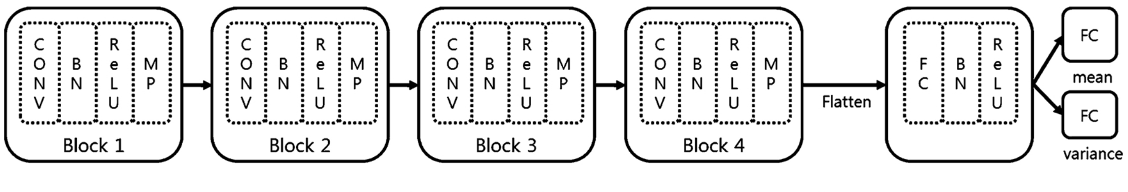
These blocks are connected in series, and after the last 4th block, the three-dimensional feature are flattened with the one-dimensional feature vector, and then one fully connected layer consisting of batch normalization and ReLU is connected. Finally, we construct two fully connected layers in parallel, one for mean and one for variance. This mean and variance are used to calculate the KL-divergence of the loss function and to sample the latent variable z.

The latent variable z is selected from the Gaussian distribution with mean μ and variance σ^2. We perform sampling from the posterior $z_i \sim q_\phi(z|x)$ using $z_i = \mu_i + \sigma_i \odot \varepsilon_i$ where $\varepsilon_i \sim N(0, 1)$. $\odot$ is an element-wise product [28]. After sampling, one-hot encoded class information is concatenated to a sampled z vector to generate images for each defect.

The decoder is symmetrically constructed with the structure of the encoder. We extend the dimension using two fully connected layers from the latent vector z, and then generate a feature map of $7 \times 7 \times 128$. We then place a block consisting of a deconvolution layer, which contains a 3×3 filter, to increase the size, and a convolution layer, batch normalization and ReLU activation layer with 3×3 filter. Like the encoder, this basic block consists of four consecutive blocks. The activation function of the last block uses a sigmoid function. The size of the final output of the decoder is the same as the input size. The label of the defect converted to one-hot encoding is concatenated with z, to serve as decoder input. For example, the one-hot encoded label of Class 1 is [1, 0, 0, 0, 0, 0], and Class 6 is [0, 0, 0, 0, 0, 1]. The detailed decoder structure is shown in Table 3.

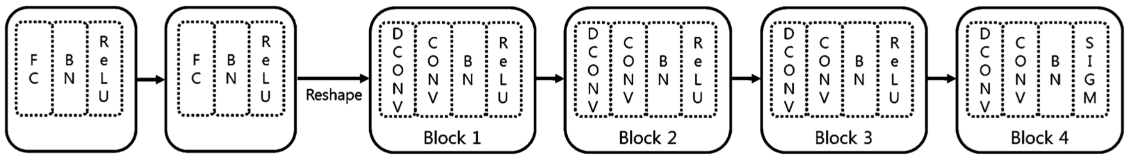
In this study, we use CCVAE to resolve issues that stem from a lack of learning data for classification model development. The training for CCVAE only uses data for training the classification models, with the exception of data used for defect classification testing. In order to compare the classification performance according to the amount of training data, as shown in Table 4, experimental conditions were divided into five cases. That is, Condition 1 is used to train CCVAE using all the training data acquired, and Condition 5 is used to train CCVAE using only 20% of the training data. To train the CCVAE, the batch size is 64, the learning rate is 0.001, and the epoch is 10,000. The dimension of the latent vector z was set to 100.

Using the learned CCVAE model, random z with normal distribution is sampled to generate an image for each defect. The defect class label to be generated is concatenated to the z of the desired image number and made into the decoder input. The output of the decoder is represented as an image, and a defect image similar to the actual defect is generated. To show an example of image generation for each class, Fig. 6 shows the generated images for condition 1. The left side of Fig. 6



CONV
BN
ReLU
MP
Block 1
Block 2
Block 3
Block 4
Flatten
FC
BN
ReLU
FC
mean
FC
variance
CONV: Convolution, BN: Batch Normalization, ReLU: Rectifier Linear Unit, MP: Max Pooling, FC: Fully Connected Layer

Fig. 4. Detailed description of the encoder in the proposed CCVAE.



CONV: Convolution, **DCONV** : Deconvolution, **BN**: Batch Normalization, **ReLU**: Rectifier Linear Unit, **FC**: Fully Connected Layer, **SIGM** : Sigmoid function

Fig. 5. Detailed description of the decoder in the proposed CCVAE.

Table 2
Structure of encoder in the CCVAE. Conv, BN, and FC mean the convolution layer, batch normalization, and fully connected layer, respectively.

Layers	Shape	Stride
Conv + BN + ReLU	3 × 3 × 1 × 16	1
Maxpooling	2 × 2 × 16 × 16	2
Conv + BN + ReLU	3 × 3 × 16 × 32	1
Maxpooling	2 × 2 × 32 × 32	2
Conv + BN + ReLU	3 × 3 × 32 × 64	1
Maxpooling	2 × 2 × 64 × 64	2
Conv + BN + ReLU	3 × 3 × 64 × 128	1
Maxpooling	2 × 2 × 128 × 128	2
Flatten	-	–
FC + BN + ReLU	6272 × 3000	–
FC FC	3000 × 100 3000 × 100	1

Table 3
Structure of decoder in the CCVAE. Deconv means the deconvolution layer.

Layers	Shape	Stride
FC + BN + ReLU	100 × 3000	–
FC + BN + ReLU	3000 × 6272	–
Reshape	–	–
Deconv	3 × 3 × 128 × 128	2
Conv + BN + ReLU	3 × 3 × 128 × 64	1
Deconv	3 × 3 × 64 × 64	2
Conv + BN + ReLU	3 × 3 × 64 × 32	1
Deconv	3 × 3 × 32 × 32	2
Conv + BN + ReLU	3 × 3 × 32 × 16	1
Deconv	3 × 3 × 16 × 16	2
Conv + BN + Sigmoid	3 × 3 × 16 × 1	1

Table 4
Number of data according to each experimental condition.

	Test data	Training data	Training data for CCVAE				
		Condition 0	1	2	3	4	5
		(raw data)	(100%)	(80%)	(60%)	(40%)	(20%)
Class1	151	151	151	121	91	60	30
Class2	152	152	152	122	91	61	30
Class3	126	127	127	101	76	51	25
Class4	88	88	88	70	53	35	18
Class5	32	32	32	27	19	13	6
Class6	106	107	107	85	64	43	21
Sum	655	657	657	526	394	263	130

is the original image for each defect and the right side is the defect image generated by CCVAE. As shown in Fig. 6, various images related to the original image are generated using CCVAE, and the generated images are used as the training data of classifier.

We use linear interpolation of the latent variable z to test the image generation according to the z change. In Fig 7 , the left and right defect images are input to the trained encoder, and latent variables z_{left} and z_{right} of each image are extracted. Using the generated z_{left} and z_{right}, a desired number of z is generated by changing the alpha value as shown in the following equation.

$$z = (1 - \alpha)z_{\text{left}} + \alpha z_{\text{right}} \tag{3}$$

where $\alpha = 0, 0.1, \ldots, 1$. Then, z is fed to the decoder to generate images in the manifold space between z_{left} and z_{right}

3.2. *Classification*

We propose a CNN-based deep learning model that can classify defects using the training data generated by CCVAE, as shown in Fig. 8. The classification model consists of four feature extraction layers and a classification layer. The feature extraction layer consists of basic blocks that contain a 3 × 3 convolution layer, ReLU activation, 3 × 3 convolution layer, ReLU activation, and drop out. At the end of the first, second, and third blocks, 2 × 2 max pooling is applied, and the size of the feature map is reduced by a factor of two. The last 4th block output is associated with classification using global average pooling (GAP), instead of max pooling. The classifier is composed of one fully connected layer. The final activation function consists of outputting a probability value between 0 and 1 using the softmax function. Cross entropy is used as loss for multiple classification. The equation of GAP and the softmax function are as follows.

$$y_j = \frac{e^{x_j}}{\sum_i^C e^{x_i}} \tag{4}$$

$$y_k = \frac{1}{\text{MN}} \sum_{i,j}^{M,N} x_{i,j,k} \tag{5}$$

The structure of the detailed classification model is shown in Table 5

4. Experimental results

The classification performance was evaluated for the defective images of the actual metal surface obtained by the proposed optical system. Experiments are conducted to verify the performance of CCVAE and the classification models according to training data conditions. As shown in Table 4, the classification models were trained for all six cases, including five wherein the training data exhibited data augmentation and one case which used only raw data, without data augmentation. To solve imbalanced data problem, image data was generated by CCVAE. As a result, the amount of training data for the classifier is equally set to 2000 in each class. The hyper parameter for learning the classification model was set as follows: the batch size is 128, the epoch is 1000, and the learning rate is 0.0001.

To measure the classification performance, the performance indexes of precision, recall ratio, accuracy and F_1 score were used. Precision and Recall are complementary, and the higher the two indicators, the better. The two indicators are defined as follows.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{6}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{7}$$

where TP is true positive, FP is false positive, and FN is false negative. It can be defined as accuracy as shown below by considering both precision and recall.

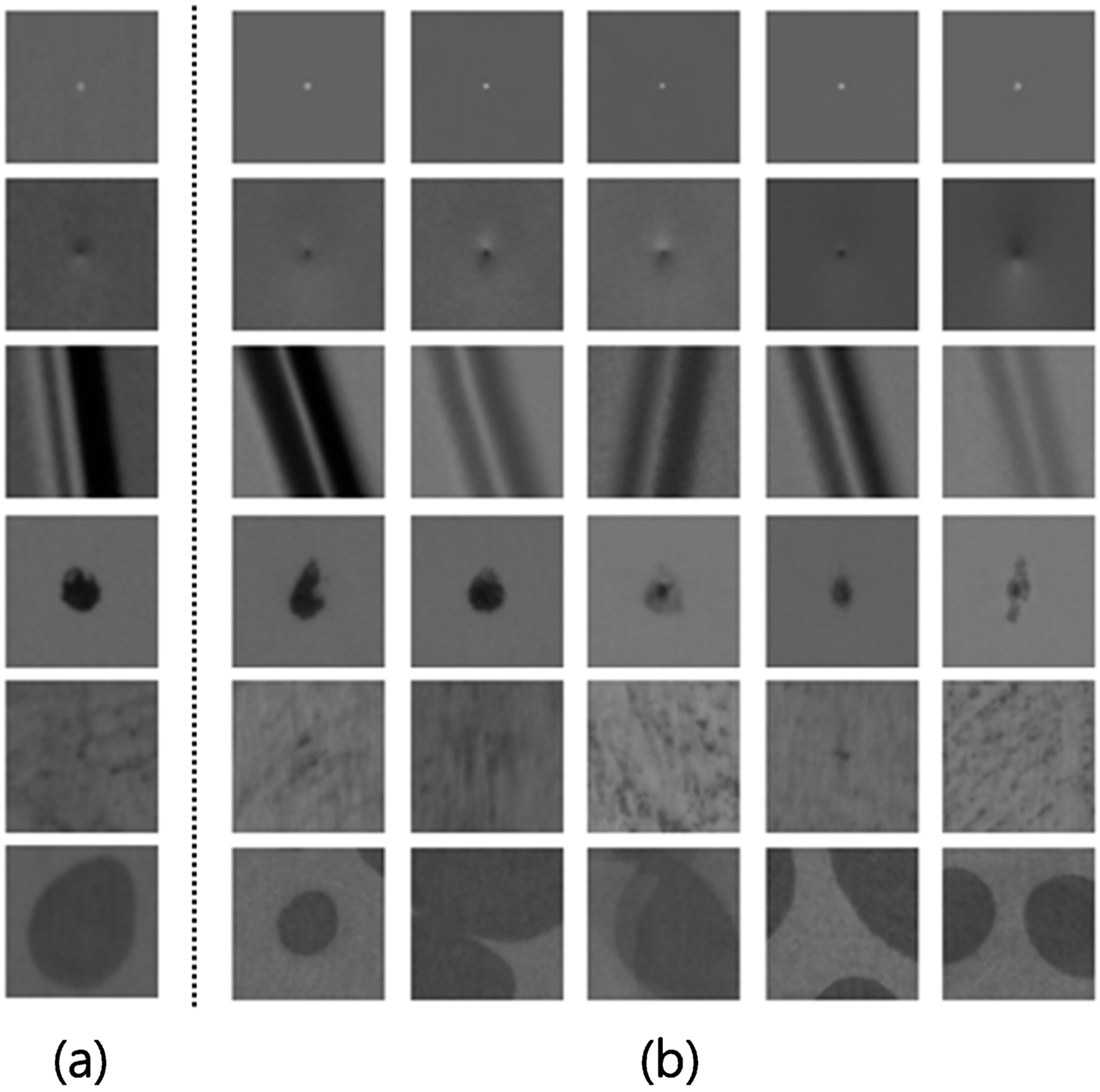


Fig. 6. Sample images: (a) original image (b) generated image by CCVAE with condition 1.

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

where TN is true negative. Accuracy is the most intuitive model performance indicator, but performance evaluation for imbalanced data, as in this study, can be evaluated more effectively when using the F_1 score. The F_1 score is a harmonic average of precision and recall rate and can be defined as follows.

$$F_1 - \text{score} = 2\frac{1}{\frac{1}{\text{precision}} + \frac{1}{\text{Recall}}} \tag{9}$$

Fig. 9 shows the change in loss up to the 100 epochs for each case. Data augmentation converges faster than when raw data is used. The performance index for the trained model's average precision, average recall rate, accuracy, and F_1- score is shown in Fig. 10. In the case of no data augmentation, the recall ratio and accuracy are 96.27% and 93.89%, respectively, but performance increases in all cases, except

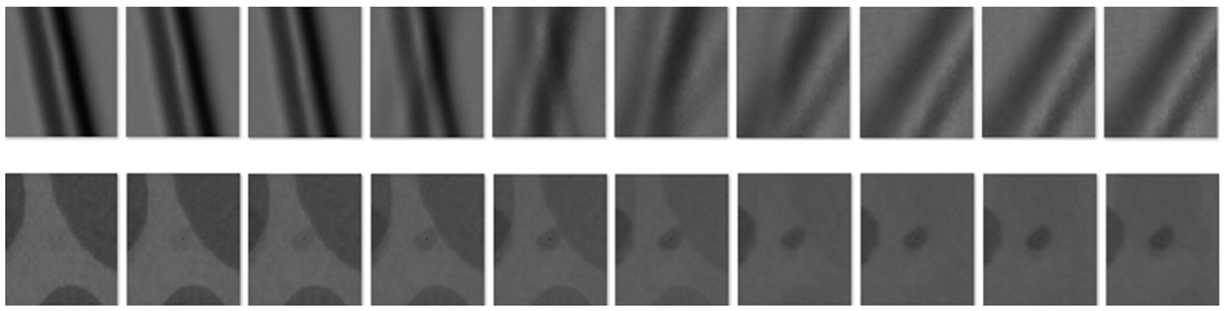


Fig. 7. Linear interpolation for latent vector. Each row is the interpolation from the left latent vector to right latent vector.

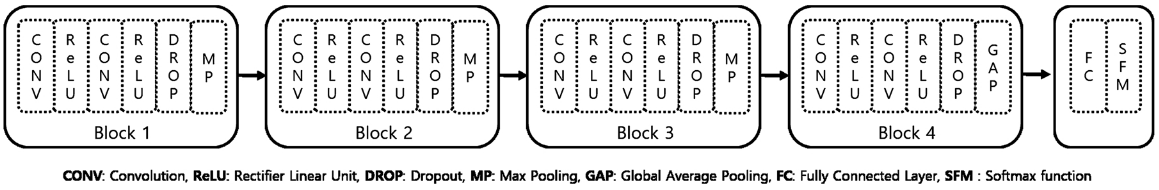

CONV
ReLU
CONV
ReLU
DROP
MP
Block 1
Block 2
Block 3
GAP
Block 4
FC
SFM
CONV: Convolution, ReLU: Rectifier Linear Unit, DROP: Dropout, MP: Max Pooling, GAP: Global Average Pooling, FC: Fully Connected Layer, SFM : Softmax function

Fig. 8. Schematic diagram of the proposed DCNN model.

Table 5
Structure of DCNN classification model.

Layers	Shape	Stride
Conv + ReLU	3 × 3 × 1 × 32	1
Conv + ReLU + Dropout	3 × 3 × 32 × 32	1
Maxpooling	2 × 2 × 32 × 32	2
Conv + ReLU	3 × 3 × 32 × 64	1
Conv + ReLU + Dropout	3 × 3 × 64 × 64	1
Maxpooling	2 × 2 × 64 × 64	2
Conv + ReLU	3 × 3 × 64 × 128	1
Conv + ReLU + Dropout	3 × 3 × 128 × 128	1
Maxpooling	2 × 2 × 128 × 128	2
Conv + ReLU	3 × 3 × 128 × 256	1
Conv + ReLU + Dropout	3 × 3 × 256 × 256	1
GAP	–	–
FC + Softmax	256 × 6	–

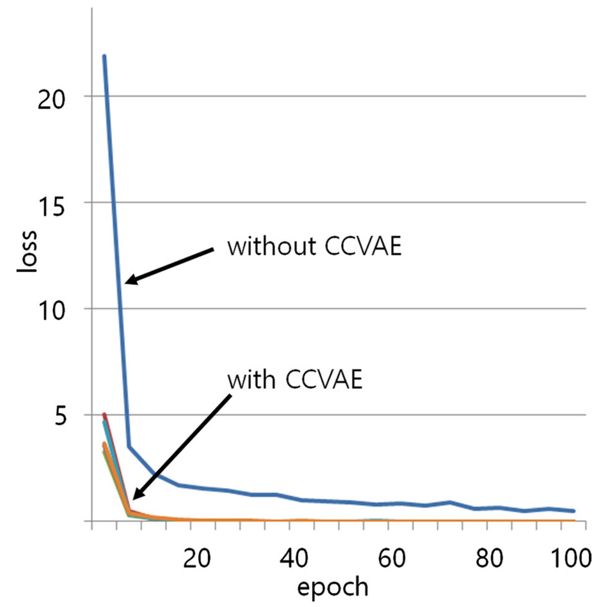


Fig. 9. Loss values up to 100 epochs for each cas.

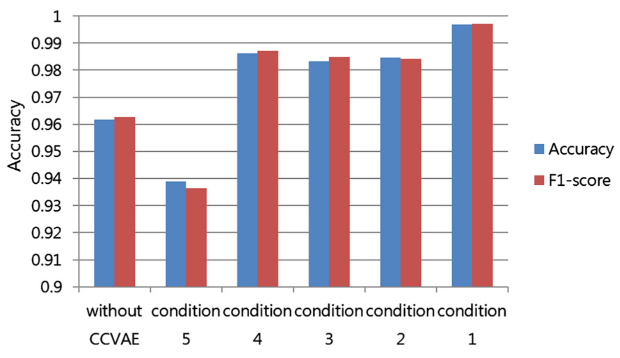


Fig. 10. Accuracy and F$_1$-scores of each experiment condition.

Table 6
Defect detection performances of the proposed method with six conditions.

Condition	1	2	3	4	5	Without CCVAE
Accuracy	0.9969	0.9847	0.9832	0.9863	0.9389	0.9618
F$_1$-score	0.9971	0.9843	0.9848	0.9871	0.9364	0.9627

Table 7
Comparision results of DCNN without CCVAE and DCNN with CCVAE.

		Predictions (DCNN without CCVAE)					
		Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
Actual values	Class 1	151	0	0	0	0	0
	Class 2	0	152	0	0	0	0
	Class 3	0	5	121	0	0	0
	Class 4	0	10	0	78	0	0
	Class 5	0	0	0	1	30	1
	Class 6	0	8	0	0	0	98
Precision		1	0.8686	1	0.9873	1	0.9899
Recall		1	1	0.9603	0.8863	0.9375	0.9245
		Predictions (DCNN with CCVAE)					
		Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
Actual values	Class 1	151	0	0	0	0	0
	Class 2	0	152	0	0	0	0
	Class 3	0	0	124	0	0	2
	Class 4	0	0	0	88	0	0
	Class 5	0	0	0	0	32	0
	Class 6	0	0	0	0	0	106
Precision		1	1	1	1	1	0.9815
Recall		1	1	0.9841	1	1	1

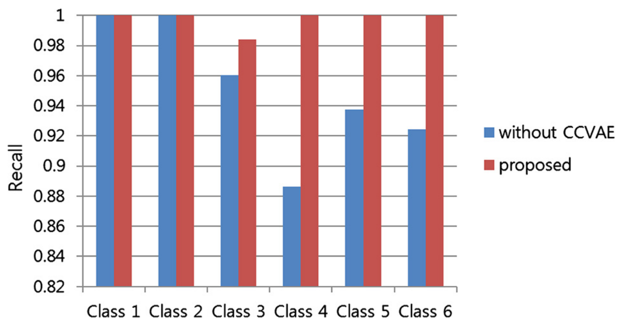


Fig. 11. Recall of each class.

Condition 5 where data augmentation is performed using only 20% data. Data generated based on extremely small amounts of data, such as Condition 5, did not minimize the distance between training and test data sets. In other words, if CCVAE is trained with a small amount of data, it is not possible to generate various forms of defects. Therefore, it does not significantly affect the improvement of generalization performance. On the other hand, in all cases except condition 5, the generalization performance is improved because more information can be extracted from the augmented data set with CCVAE than the data set without CCVAE. When data augmentation was not performed using CCVAE, the accuracy was 96.27%. But in the case of data augmentation using CCVAE, it increased at rates from 3.42% to 99.69%, as shown in Table 6. The F$_1$-score also increased from 96.27%, up to 99.71%. It is

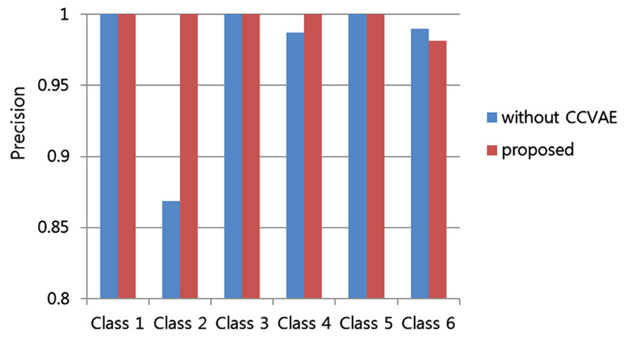


Fig. 12. Precision of each class.

important to have at least 99% accuracy and F1-score because the classification results should be used to determine the grade of the final product.

Table 7 compares the performance of each defect class. Table 7 shows the confusion matrices for Condition 1 with best classification performance and without data augmentation. The first column represents the class labels of the test data, and the first row shows the classification results of the deep learning model. For example, if the first confusion matrix is analyzed in the row direction, the number 78 in Class 4 indicates that 78 out of 88 test data were correctly classified as Class 4. And number 10 on the left indicates the number of images classified as Class 2 incorrectly. On the other hand, if we analyze the confusion matrix in the column direction, it indicates that among the 79 results classifying the deep running model as Class 4, the actual Class 4 is 78 and the remaining 1 are misclassified.

Defect Classes 4, 5, and 6 have fewer data than defect Classes 1, 2, and 3. Therefore, the performance of both recall ratio and precision is relatively poor when learning without CCVAE, as shown in Table 7. This overall performance degradation is due to a lack of data and an imbalanced data. However, in the case of data augmentation with CCVAE, performance degradation owing to the amount of data was resolved, and the performance was excellent for all classes. Figs. 11 and 12 compare the recall and precision of each confusion matrix for each class. In the figure, it is seen that the performance of the recall for Classes 4, 5, and 6, with a relatively small amount of data, is remarkably increased after applying data augmentation using CCVAE. Therefore, the defect classification system proposed in this study showed good performance when tested on actual metal surface defects with imbalanced data.

5. Conclusion

In this paper, a new deep neural networks-based defect inspection system for metal surfaces is presented. A novel CNN-based architecture, which is a data-driven approach, is proposed for the purpose of automatically classifying defects in metal surfaces. In order to obtain excellent performance in such a deep CNN model, a large quantity of learning data is required, and in particular, the amount of data for each class should be similar. However, owing to the nature of the manufacturing industry, there are many cases where the incidence per defect differs. To solve this imbalanced data problem, this study proposed a data augmentation method based on a variational autoencoder. To improve the image generation performance, a network is constructed as a convolution layer without using a fully connected layer. We also proposed a conditional VAE method that concatenated the class condition to the latent variables, to generate data for each defect. To test the data augmentation effect of the CCVAE, according to the amount of learning data, and to the performance of the classification model using generated data, we experimented with varying amounts of training data. Experiments were performed on the surface images obtained on actual metal production lines. The experimental results show that the proposed algorithm demonstrates good detection performance.

Acknowledgment

This study has been conducted with the support of the Korea Institute of Industrial Technology.

References

[1] Kuo CFJ, Lo WC, Huang YR, Tsai HY, Lee CL, Wu HC. Automated defect inspection system for CMOS image sensor with micro multi-layer non-spherical lens module. J Manuf Syst 2017;45:248–59.
[2] Wang J, Fu P, Gao RX. Machine vision intelligence for product defect inspection based on deep learning and Hough transform. J Manuf Syst 2019;51:52–60.
[3] Zhang B, Yang H, Yin Z. A region-based normalized cross correlation algorithm for the vision-based positioning of elongated IC chips. IEEE Semicond Manuf 2015;28:345–52.
[4] Tong L, Wong WK, Kwong CK. Fabric defect detection for apparel industry: a nonlocal sparse representation approach. IEEE Access 2017;5:5947–64.
[5] Raheja JL, Ajay B, Chaudhary A. Real time fabric defect detection systems on an embedded DSP platform. Opt-Int J Light Electron Opt 2013;124:5280–4.
[6] Oh JH, Yun BJ, Kim SY, Park KH. A Development of the TFT-LCD Image Defect Inspection Method Based on Human Visual System. IEICE Trans Fundam Electron Commun Comput 2008;E91-A:1400–7.
[7] Arnal L, Solanes JE, Molina J, Tornero J. Detecting dings and dents on specular car body surfaces based on optical flow. J Manuf Syst 2017;45:306–21.
[8] Li X, Jiang H, Yin G. Detection of surface crack defects on ferrite magnetic tile. NDT & E Int 2014;62:6–13.
[9] Prasanna P, Dana KJ, Gucunski N, Basily BB, La HM, Lim RS, et al. Automated Crack Detection on Concrete Bridges. IEEE Trans Automat Sci Eng 2016;13:591–9.
[10] Pham VH, Lee BR. An image segmentation approach for fruit defect detection using k-means clustering and graph-based algorithm. Vietnam J Comput Sci 2015;2:25–33.
[11] Hanzaei SH, Afshar A, Barazandeh F. Automatic detection and classification of the ceramic tiles surface defects. Pattern Recog 2017;66:174–89.
[12] Tsai DM, Molina DER. Morphology-base defect detection in machined surfaces with circular tool-mark patterns. Measurement 2019;134:209–17.
[13] Loizou J, Tian W, Robertson J, Camelio J. Automated wear characterization for broaching tools based on machine vision systems. J Manuf Syst 2015;37:558–63.
[14] Tootooni MS, Liu C, Roberson D, Donovan R, Rao PK, Kong ZJ, et al. Online non-contact surface finish measurement in machining using graph theory-based image analysis. J Manuf Syst 2016;41:266–76.
[15] Zhang H, Jin X, Wu MJ, Wang Y, He Z, Yang Y. Automatic visual detection system of railway surface defects with curvature filter and improved Gaussian mixture model. IEEE Trans Instrumentation Measure 2018;67:1593–608.
[16] Zhao J, Peng Y, Yan Y. Steel surface defect classification based on discriminant manifold regularized local descriptor. IEEE Access 2018;6:71719–31.
[17] Park C, Choi S, Won S. Vision-based inspection for periodic defects in steel wire rod production. Opt Eng 2010;49:017202.
[18] Kang D, Jang YJ, Won S. Development of an inspection system for planar steel surface using multispectral photometric stereo. Opt Eng 2013;52:039701.
[19] Choi DC, Jeon YJ, Kim SH, Moon S, Yun JP, Kim SW. Detection of pinholes in steel slabs using Gabor Filter combination and morphological Features. ISIJ Int 2017;57:1045–53.
[20] Xi J, Shentu L, Hu J, Li M. Automated surface inspection for steel products using computer vision approach. Appl Opt 2017;56:184–92.
[21] Jhang Y, You D, Gao X, Zhang N, Gao P. Welding defects detection based on deep learning with multiple optical sensors during disk laser welding of thick plates. J Manuf Syst 2019;51:87–94.
[22] Tao X, Zhang D, Ma W, Liu X, Xu D. Automatic metallic surface defect detection and recognition with convolutional neural networks. Appl Sci 2018;8:1575.
[23] Yi L, Li G, Jiang M. An end-to-end steel strip surface defects recognition system based on convolutional neural networks. Steel Res Int 2017;88:1600068.
[24] Zhou S, Chen Y, Zhang D, Xie J, Zhou Y. Classification of surface defects on steel sheet using convolutional neural networks. Material Tehnol 2017;51:123–31.
[25] Pawara P, Okafor F, Schomaker L, Wiering M. Data Augmentation for Plant Classification. International Conference on Advanced Concepts for Intelligent Vision Systems 2017;61:5–62. 6.
[26] Zhu X, Liu Y, Qin Z, Li J. Emotion classification with data augmentation using generative adversarial networks. 2017 [Online]. Available: https://arxiv.org/pdf/1711.00648.
[27] Sohn K, Lee H, Yan X. Learning structured output representation using deep conditional generative models. in Advances in Neural Information Processing Systems 2015:3483–91.
[28] Kingma DP, Welling M. Auto-encoding variational bayes. 2013 [Online]. Available: https://arxiv.org/pdf/1312.6114.
[29] Doersch C. Tutorial on variational autoencoders. 2016 [Online]. Available: https://arxiv.org/pdf/1606.05908.
[30] Semeniuta S, Severyn A, Barth E. A hybrid convolutional variational autoencoder for text generation. Conference on Empirical Methods in Natural Language Processing. Denmark: Copenhagen; 2017. p. 627–37.
[31] An J, Cho S. Variational Autoencoder Based Anomaly Detection Using Reconstruction Probability. Technical report. Seoul. Korea: SNU Data Mining Center; 2015.